Bioinformatikai eredetű kombinatorikai problémák

Erdős Péter

2006

ÉRTEKEZÉS

az MTA Doktora cím elnyerésére

Tartalomjegyzék

A csatolt cikkek listája

L.A. Székely - M.A. Steel - P.L. Erdős: Fourier calculus on evolutionary trees, *Advances in Appl. Math* **14** (1993), 200–216.

P.L. Erdős - L. A. Székely: Counting bichromatic evolutionary trees, *Discrete Appl. Math.* **47** (1993), 1–8.

P.L. Erdős - L. A. Székely: On weighted multiway cuts in trees, *Mathematical Programming* **65** (1994), 93–105.

P.L. Erdős - A. Frank - L.A. Székely: Minimum multiway cuts in trees, *Discrete Appl. Math.* **87** (1998), 67–75.

P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule, *Computers and Artificial Intelligence* **16** (1997), 217–227.

P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: A few logs suffice to build (almost) all trees (I), *Random Structures and Algorithms* **14** (1999), 153–184.

P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: A few logs suffice to build (almost) all trees (II), *Theoretical Computer Science*, **221** (1-2) (1999), 77–118.

P.L. Erdős - P. Ligeti - P. Sziklai - D.C. Torney: Subwords in reverse complement order, in press *Annals of Combinatorics* **10** (2006) 415–430.

Tárgymutató

Bevezetés

A disszertáció 1990-óta keletkezett, alapvetően bioinformatikai eredményeket ismertet: a problémák döntő többsége a molekuláris biológia jelenlegi forradalmában felmerült kombinatorikai kérdésekből ered.

Alkalmazott problémáknál gyakran előfordul, hogy a megoldhatóság kedvérért az alkalmazott matematikai modellt olyan mértékig kell egyszerűsiteni, hogy az eredmények már nem is igazán hasznosak az eredeti problémák szempontjából. Az is gyakran előfordul, hogy bár a rendelkezésre álló eszközökkel kezelhető feladatok hasznosak, de matematikai értelemben már érdektelenek: megoldásuk könnyű vagy elméleti szempontokból nem mondanak újat.

Meggyőződésem szerint az ebben a disszertációban tárgyalt kérdések nem ilyenek: a nyert tételek, eljárások és algoritmusok a gyakorlatban hasznosak, jól alkalmazhatók, ugyanakkor matematikailag is érdekesek, mert tisztán matematikai problémaként önállóan is megállják a helyüket.

A dolgozatban szereplő eredmények jelentős része hosszú (esetenként bonyolult) bizonyítással bír, ezek többségét itt nem ismertetem. Ehelyett a fő súlyt a felmerült matematikai problémák hátterét (avagy jogosultságát) szolgáltató biológiai modellek matematikusok számára érthető kifejtésére helyezem. Azaz a diszszertáció "rövid értekezés" formájában került megírásra: egy, a szokásosnál hosszabb bevezető után a releváns cikkek mellékletként szerepelnek benne.

A dolozatban három fő rész található, összesen kilenc szakaszból áll, továbbá nyolc cikk szerepel mellékletként. A első két részben un. *evolúciós fákat* vizsgálok. Ezek (gyakran gyökeres) bináris fák, melyek levelei egy-egy értelműen címkézettek, míg belső (elágazó) csúcsaik nem. A biológusok ezeket használják a fajok közötti leszármazási kapcsolatok ábrázolására (és megtalálására). A biológiai adatokat kevés (tipikusan 2, 4 vagy 20) szín felhasználásával alkotott színvektorok hordozzák, továbbá a fával ábrázolt történések valamilyen biológusok által feltételezett modell szerint történnek.

Az első részben ez a modell a statisztikából ismerős parsimonia elv. Az itt felmerülő optimalizációs problémák általában legalább duplán exponenciálisak, pontos megoldásukra kevés a remény. Ezért az előállított modellfák közül gyakran statisztikai alapon választanak "megfelelőt". Ebben a részben ilyen statisztikákkal kapcsolatos kombinatórikai problémákat vizsgálunk. Közülük az első egy leszámlálási kérdés, amely megoldása a jól ismert Menger tételeken alapuló dekompozíciót használ. A módszerek kettőnél több színre történő alkalmazásához a *multiway cut* probléma jobb megértése lehet

szükséges, amely az első rész másik témája.

A dolgozat második része evolúciós fák néhány sztochasztikus modelljével foglalkozik. Részben mutatószámokat illetve eszközöket fejleszt ki a modellek illetve módszerek összehasonlítására, részben pedig gyors algoritmusokat ad egy modellosztályban a helyes evolúciós fák 1 valószínűségű megtalálásához.

A disszertáció harmadik része véges ábécé feletti korlátos hosszúságú szavak rész-szavakból történő rekonstrukcióját vizsgálja, amely microarray kisérletek illetve úgynevezett *DNS kódok* tervezéséhez nyújthat segítséget.

1. A multiway cut probléma

A modern kombinatorikus optimalizálás egy sokat vizsgált területe a *multiway cut* probléma: adott a G gráf élein egy w súlyfüggvény. Adott továbbá *terminál pontok* egy k elemű halmaza. Keressünk minimális összsúlyú élvágást, ami a terminál pontokat páronként szeparálja: az élek elhagyásával keletkezett gráfban különféle színű pontok között nincsenek utak. A $k = 2$ eset a klasszikus él-Menger probléma. Mint a Dahlhaus - Johnson - Papadimitriou - Seymour - Yannakakis cikk ([DahJoh92]) bebizonyítja, a probléma NP-nehéz még a legegyszerűbb esetben is (három szín, egység súly). Ugyanebben a cikkben található az első approximáló algoritmus a problémára. Szintén itt bizonyítják be, hogy síkgráfokon a probléma kezelhető polinomiális időben, ha a színek száma korlátos. A probléma, különösen az utóbbi tíz évben, komoly kutatásokat indukált, számos eredménnyel.

Székely Lászlóval közös cikkeinkben ([1, 2, 7, 10, 13]) bevezettük az eredeti multiway cut probléma egy általánosítását: legyen $G = (V, E)$ egy egyszerű gráf, $C = \{1, 2, \ldots, r\}$ pedig egy színhalmaz. Ha $N \subseteq V(G)$ a terminál pontok halmaza, akkor egy $\chi : N \to C$ leképezést *parciális színezés*-nek hívunk. Ekkor egy $\bar{\chi} : V(G) \to C$ leképezést akkor mondunk *színezésnek*, ha a két leképezés megegyezik a terminál pontokon. Az *általánosított multiway cut* probléma egy olyan legkisebb súlyú élrendszer megtalálása, amely bármely két, eltérő színű terminál pontot szeparál.

Amint azt Dahlhaus - Johnson - Papadimitriou - Seymour - Yannakakis cikkeikben ([DahJoh92, DahJon94]) kimutatják, bár az általánosított multiway cut tetszőleges gráfokon megegyezik az eredeti multiway cut problémával, speciális gráfosztályokon azonban (mint síkgráfokon vagy acyclikus gráfokon) eltérőek. Például síkgráfokon az általánosított multiway cut már három szín mellett és egységsúlyú élekkel is NP-teljes ([DahJoh92]).

A cikkekben bevezettünk egy új típusú alsó korlátot a multiway cut súlyára, továbbá egy új típusú pakolási feladat felhasználásával illetve egy minimax tétel bebizonyításával teljesen megoldottuk a fák multiway cut problémáját. Ennek részben elméleti következményei vannak (lásd például [DahJon94]), továbbá az evolúciós fák elméletében is felhasználásra kerültek (például [PenLoc94]). Az multiway cut-nak párhuzamos SQL-lekérdesések tervezése témakörében is vannak alkalmazásai (például [HasMan98]), továbbá kommunikációs hálózatok elméletében (például [Pou06]). Ez utóbbi dolgozat a kommunikációs költségek minimalizálásával foglalkozik szétosztott processzor hálózatok esetén. Kimutatja, hogy a feladat leírásához az általunk bevezetett általánosított multiway cut probléma az alkalmas, majd a "partial distribution problem" megoldására a színfüggű súlyfüggvényre kialakított algoritmusunkat alkalmazza.

1.1. Minimális súlyú színezések

A (számunkra fontos) biológiai alkalmazásokban a konstans élsúlyoknál bonyolultabb súlyfüggvényekre van szükség . Ehhez jelölje $E(G) \times 2$ a gráf irányított éleit (azaz mindegyik él mindkét irányítással jelen van). Egy $W : E(G) \times 2 \to \mathbb{N}^{r \times r}$ leképezés egy (színfüggő) *súlyfüggvény,* ha a $W(p,q)$ és $W(q,p)$ mátrixok megegyeznek, továbbá a főátlókban csupa nulla van. A ${}_iW(p,q)_j = w(p,q;i,j)$ elem azt mondja meg, hogy a (p,q) élnek mennyi a súlya egy $\bar{\chi}$ színezésben, ha $\bar{\chi}(p) = i, \bar{\chi}(q) = j$ (avagy $\bar{\chi}(p) = j, \bar{\chi}(q) = i$, ami ugyan azt az értéket adja). A W színfüggetlen, ha minden főátlón kívüli elem azonos. A súlyfüggvény értelemszerűen lesz élfüggetlen. Végül W *konstans*, ha egyszerre szín- és élfüggetlen. Bármely χ parciális színezés partícionálja a terminál pontokat: az azonos színű pontok kerülnek azonos osztályba. Ebben a gráfban élek egy halmaza, amelyek együtt bármely két, eltérő színű terminál pontot elválasztanak, egy *multiway cut*-ot alkot. Világos, hogy egy $\bar{\chi}$ színezés színváltó élei mindig multiway cut-ot alkotnak. Egy $\bar{\chi}$ színezés súlya a színváltó élek összsúlya. Az adott gráfon egy χ parciális színezés $\ell(G,\chi)$ hossza az összes lehetséges színezés súlyának a minimuma.

A $\ell(G,\chi)$ mennyiség meghatározásának komplexitása függ a súlyfüggvény és a gráf szerkezetétől. Biológiai alkalmazásokban a gráfok általában címkézett levelekkel és nem-címkézett belső pontokkal rendelkező bináris fák, ahol a parciális színezés a leveleken adott. Ezeket az objektumokat hívják *evolúciós fáknak*. Konstans súlyfüggvények esetén evolúciós fákra W.M. Fitch dolgozott ki először egy lineáris algoritmust a hosszúság meghatározására. (Az

algoritmus korrekt volt, bár a biológus Fitch ezt nem látta szükségesnek bizonyítani. Ezt először a matematikus Hartigan tette meg.) Székely Lászlóval közös [1] cikkünkben szintén adunk egy (a korábbiaktól különböző) bizonyítást az algoritmus helyességére.

A Székely Lászlóval közös [10] cikk tetszőleges, levél színezett fákra ad unárisan polinomiális algoritmust színfüggő súlyfüggvény esetén a hossz meghatározására. (Itt minden egyes numerikus adatot egy-egy számnak tekintünk, függetlenül annak nagyságától, azaz attól, hogy milyen módon ábrázolja a számítógép.) Az algoritmus arra is alkalmas, hogyha minden belső pontban megadunk egy megendegett színhalmazt, akkor az algoritmus valamelyik megengedett színt rendeli a belső pontokhoz is. (Arra azonban nincs esély, hogy polinomiális időben megkeressük az összes optimális színezést, mert ebből akár exponenciálisan sok is lehet - mint azt M.A. Steel egy eredménye megmutatta.)

A cikk egyébként ennél egy kicsit általánosabb állítást igazol:

1.1. Tétel ([10] Section 3**).** *Legyen a gráf olyan, amelynek minden körét a terminál pontok lefedik. Ekkor létezik unárisan polinomális algoritmus egy optimális színezés meghatározására színfüggetlen súlyfüggvény esetén.*

Korábban Sankoff és Cedergen illetve Williamson és Fitch élfüggetlen (de színfüggő) súlyfüggvényeket tanulmányoztak, és közreadtak különféle gyors, bár csak heurisztikus algoritmusokat (azaz nem vizsgálták az algoritmusuk helyességét vagy igazi futásigényét).

Lényegesen bonyolultabb kérdést kapunk, ha levelek egy adott L halmazához és a rajtuk adott χ parciális színezéshez meg akarjuk határozni az összes, a levelekre illeszkedő bináris fa közül azt, amelyiknek a legkisebb a hossza a χ-re nézve. Ha a leveleket ma élő fajok alkotják, és a színezés pedig valamilyen biológiai jellemzőjüket jelenti (például morfológiai jegyek, vagy az átörökítő anyag egy jellemző része), akkor a legrövidebb fa megtalálása azt a nézetet testesíti meg, hogy a természet az élet kialakításánál takarékos volt, a lehető legkevesebb változást használta fel az összes létező élőlény kialakításához. Ezt *parsimonia elvnek* hívják, és tipikus feltevés különböző statisztikai vizsgálatoknál.

Az evolúció kutatói ezeket a biológiai jellemzőket *karakter*-eknek hívják. Azaz az i-ik karakter matematikai értelemben a színvektor i-ik koordinátáját jelenti.

A valós helyzetekben, azaz létező biológiai rendszerek vizsgálatakor, persze nem csak egyetlen jellemző ír le egy-egy fajt, ezért minden fajt (azaz

a keresett bináris fa leveleit) hosszabb színvektorok jellemeznek. Annak eldöntése, hogy ilyen színvektorok esetén létezik-e pontosan k hosszúságú fa a χ parciális színezésre nézve (ilyenkor az adott fára minden koordinátában külön kiszámoljuk a hosszat, majd összeadjuk) NP-nehéz feladat, ezért az érdekes gyakorlati esetekben ezt lehetetlen eldönteni. Ez egyébként Graham és Foulds egy eredménye [GraFou82]. Ezért a parsimoniával foglalkozók egyik fő célnak az evolúciós fák statisztikai tulajdonságainak meghatározását tartják. Ezt úgy lehetséges felhasználni egyes keresett evolúciós fák rekonstrukciójánál, hogy az éppen vizsgált algoritmus "termékeit" a statisztikailag elvárható fákkal hasonlítják össze. Minél közelebb van az elvárhatóhoz, annál jobb. Ezen statisztikai vizsgálatok egyik lehetséges lépése az adott levélszínezéshez tartozó, éppen k hosszúságú fák leszámlálása.

A legegyszerűbb eset megtárgyalásához rögzítsünk egy adott egy-karakteres, azaz egy hosszú színvektorokból álló 2-színezést az L levél halmazon. Legyen a és b a két színosztály mérete. Mennyi azon evolúciós fák $f_k(a,b)$ száma, amelyek hossza az adott levélszínezés mellett éppen k. A választ erre Carter és munkatársai (1990)-ben adták meg:

Tétel. [Carter - Hendy - Penny - Székely - Wormald: ([CarHen90])]

$$f_k(a,b) = (k-1)!(2n-3k)N(a,k)N(b,k)\frac{b(n)}{b(n-k+2)}$$

ahol $a+b=n$, $a>0$, $b>0$, és ahol $N(x,k)$ jelöli az összesen x levéllel rendelkező és k darab evolúciós fából álló erdők számát.

(A [9] cikkem, egyebek között, egy bijektív bizonyítást adott az $N(x,k)$ mennyiségekre.) A Carter tételre az eredeti bizonyítás többváltozós Lagrange inverziót és computer algebrát alkalmazott. M.A. Steel talált egy jobb, bijektív megközelítést ([Steel93]), amire Székely Lászlóval közös [7] cikkünkben adtunk viszonylag rövid és transzparens bizonyítást. A módszer legfőbb érdekessége, hogy a leszámlálás előtt bebizonyítja a k hosszú evolúciós fák egy struktúra tételét, amely eredmény az él-Menger és a pont-Menger tételek felváltott alkalmazásain alapul.

A kettőnél több színnel színezett evolúciós fák leszámlálásához szükség lenne az evolúciós fákra vonatkozó analóg tételek bebizonyítására. A több színű pont-Menger tétel fákra változtatás nélkül teljesül, de ugyanez az él-Menger (azaz a multiway cut) problémára nem igaz.

1.2. Egy minimax eredmény fák multiway cut problémájára

Mivel az általánosított multiway cut probléma már $k = 3$ esetben is NP-nehéz, természetesen nem lehet elvárni általánosan érvényes, a Menger tételhez hasonló minimax eredményt vele kapcsolatban. Valóban, mint az közismet, már a $k = 3$ esetben sem igaz az él-Menger tétel analógja: egyszerű ellenpélda rá az egység élsúlyokkal ellátott, a leveleket terminál pontokként tartalmazó $K_{1,3}$ csillag. Az előző szakaszban említett leszámlálási feladat kettőnél több színre történő analóg megoldásához szükség lenne egy fákra érvényes minimax tétel bebizonyítására. Egy ilyet a [1, 2, 10] cikksorozatban sikerült Székely Lászlóval közösen kimunkálnunk. Megjegyzendő, hogy ennek felhasználásával M.A. Steel valóban tovább lépett a leszámlálási feladat tárgyalásában ([Steel93]).

A [1] cikkben a súlyozatlan esettel foglalkoztunk (pontosabban szólva itt minden él súlya 1), míg a [2, 10] dolgozatokban színfüggetlen súlyfüggvények esetére dolgoztuk ki a megfelelő minimax eredményt. A szakasz hátralévő részében irányítatlan gráfokban, két-két terminál pont közé, *irányított (oriented)* utakat pakolunk. Irányított út úgy keletkezik egy iranyítatlan P útból, hogy megmondjuk, hogy a határoló terminál pontok közül melyik az $s(P)$ kezdő pont, és melyik a $t(P)$ végpont, továbbá feltesszük, hogy az utak nem érintenek más terminál pontot.

1.2. Definíció. *Egy út akkor* színváltó*, ha χ szerint eltérő színű terminál pontok között fut. Két színváltó út* konfliktusban van*,*

(a) *ha egy adott élt ellenkező irányban használnak (az utak irányítását tekintve),*

(b) *ha két út ugyan azonos irányban használ egy élt, de végpontjaik színe χ szerint megegyezik.*

Ekkor a [1] cikk szerint következő alsó becslés teljesül a multiway cut nagyságára:

1.3. Tétel. *Legyen G hurokél mentes, irányítatlan gráf terminál pontok egy N halmazával és egy χ parciális színezéssel. Legyen továbbá $\mathcal{P}$ irányított utak egyrendszere a terminál pontok között, hogy semelyik kettő nincs konfliktusban. Ekkor $|\mathcal{P}|$ sohasem nagyobb, mint bármely G-beli multiway cut elemszáma.*

Ha egy gráfban a terminál pontok N halmaza lefed minden kört, akkor minden egyes N-beli pontot vágjunk annyi példányra, amennyi a foka, és minden példány színe legyen megegyező a pont eredeti χ szerinti színével. A keletkezett objektum ekkor egy levél-színezett fa. Ez az egyszerű eljárás az alapja, hogy az [1] cikknek az eredetileg fák multiway cut problémáját megoldó minimax tétele a következő kicsit általánosabb formában is kimondható:

1.4. Tétel. *Legyen G hurokél mentes, irányítatlan gráf, terminál pontok egy N halmazával, amit egy χ parciális színezés k színnel színez meg. Tegyük fel, hogy N pontjai a G minden körét lefedik. Ekkor, ha irányított utak egy $\mathcal{P}$ rendszere olyan, hogy semelyik két út sincs konfliktusban, akkor az útrendszer számossága megegyezik a legkisebb multiway cut elemszámával.*

A tétel bizonyítása a megkívánt útrendszer rekurzív megkonstruálásán alapul. Az algoritmus futásideje polinomiális.

Vegyük észre, hogy miután a keresett útrendszer semelyik két eleme sincs konfliktusban egymással, ezért az utak a fa felhasznált élein egyértelműen meghatároznak egy irányítást. Van-e mód ennek az irányításnak a meghatározására az útrendszer rögzítése nélkül?

A kérdésfeltevés mögött az a gondolat, hogyha sikerül megtalálni az említett irányítást, akkor már a szokásos él-Menger tétel k-szoros alkalmazásával meg lehet határozni az útrendszert. Nevezetesen egy színt elkülönítünk az összes többitől, és az irányított gráf ebben a 2-színezésében keresünk irányított utakat.

A vázolt gondalatmenetet a Frank Andrással és Székely Lászlóval közös [13] cikkben sikerült bizonyítássá érlelni. (Megjegyezzük, hogy a következőkben a parciális színezés terminál pontok egy S halamzát színezi, méghozzá úgy, hogy minden szín egy ponton fordul elő. Ha nem ez a helyzet, akkor minden színre az összes azonos színű pontot egyesítjük. Továbbá mostantól a multiway cut méretét π_S-sel jelöljük.) Először is szükségünk van néhány további definícióra:

Legyen $\vec{G}$ egy irányított gráf, legyen Z csúcsok egy részhalmaza. Ekkor legyen $\varrho_{\vec{G}}(Z)$ a $\vec{G}$-ben a Z ponthalmazba belépő élek száma ("befok"). Továbbá az A, B diszjunkt ponthalmazokra legyen $\lambda(A, B; \vec{G})$ az A-ból induló, B-ben végetérő, páronként éldiszjunkt irányított utak maximális száma. Az él-Menger tétel szerint ekkor $\lambda(A, B; \vec{G}) = \min(\varrho(X) : B \subseteq X \subseteq V - A)$. A G hurokél mentes gráfra és az $s \in S \subseteq V(G)$ pontra legyen $\lambda(S \setminus s, s; G)$ az $(S \setminus s)$ és az s között futó éldiszjunkt utak maximális száma. Jelölje

$\lambda(S-s,s;\vec{G})$ ugyanezt az irányított gráfban, irányított utakkal. A Menger tétel alapján mindkét mennyiség polinomiális kiszámítható.

Lovász László vezette be a $\tau_S^* := \sum_{s\in S}\lambda(S-s,s;G)/2$ mennyiséget, frakcionális S-útpakolásokkal kapcsolatban. Egy további mennyiség egy G-beli T részfa *értéke*, amely a benne levő S-beli pontok száma, mínusz 1. Legyen ν_S^{tree} a G-beli páronként éldiszjunkt részfák értékei összegének a maximuma. Végezetül legyen $\vec{\nu}_S := \max\left(\sum_{s\in S}\lambda(S-s,s;\vec{G})\right)$, ahol $\vec{G}$ végigfut a G lehetséges összes irányításán. Ekkor

1.5. Tétel ([13] Theorem 1.1**).**

$$\tau_S^* \le \nu_S^{tree} \le \vec{\nu}_S \le \pi_S. \tag{1}$$

Megjegyzendő, hogy a $\vec{\nu}_S$ éppen az olyan irányított S útrendszerek maximális mérete, hogy semelyik két irányított út ne legyen konfliktusban egymással. Ezután a cikkben bebizonyítjuk a 1.4. Tétel következő változatát:

1.6. Tétel ([13] Theorem 2.1**).** *Legyen $G=(V,E)$ egy hurokél mentes gráf, terminál pontok egy S halmazával, ahol $G-S$ egy fát indukál. Ekkor a minimális multiway cut*

$$\vec{\nu}_S = \max\sum_{s\in S}\lambda(S-s,s;\vec{G}) \tag{2}$$

ahol a maximalizálás az összes lehetséges $\vec{G}$ irányításon fut.

A tétel bizonyításában a gráf szükséges irányítása rekurzív módon, polinomiális időben kerül meghatározásra.

A következőkben a Székely Lászlóval közös [10] cikk alapján vázolom hurokél mentes gráfok tetszőleges, azaz él- és színfüggő, súlyozása mellett egy lehetséges alsó becslést a (súlyozott) multiway cut értékére, és bemutatok egy, a 1.4. Tétellel analóg minimax eredményt fák súlyozott multiway cut problémájára.

Legyen G hurokél mentes gráf terminál pontok egy N halmazával, ahol a parciális színezés megint k színt használ . Legyen $\mathcal{P}$ színváltó irányított N utak halmaza (egyetlen út sem tartalmaz N-beli belső pontot, de valamely út több példányban is jelen lehet). Legyen továbbá $e=(p,q)\in E(G)$ egy rögzített él. Ekkor legyen

$$n_i(e,\mathcal{P}) = \#\{P\in\mathcal{P} \ : \ (p,q)\in P \text{ és } \chi(t(P))=i\},$$

ahol a $t(P)$ újra az illető út végpontját jelöli, a $(p,q) \in P$ jelölés pedig azt jelenti, hogy az út a p pontban lép be az élbe, és a q pontban hagyja el az élt. Ezután színváltó utak egy rendszerét *útpakolás*nak mondjuk, ha minden $i \neq j$ színpárra és minden (p,q) élre teljesül:

$$n_i((p,q),\mathcal{P}) + n_j((q,p),\mathcal{P}) \leq w(p,q;j,i).$$

Jelölje $p(G,\chi)$ a lehetséges útpakolások maximális, multiplicitásos elemszámát. Ekkor

1.7. Tétel ([10] Theorem 1**).** *Legyen G tetszőleges, hurokél mentes gráf az N terminál halmazzal és a χ parciális színezéssel. Legyen W egy (színfüggő) súlyfüggvény a gráfon. Ekkor teljesül:*

$$\ell(G,\chi) \geq p(G,\chi).$$

Teljesül továbbá a következő minimax tétel is (a súlyfüggvény itt kevésbé általános):

1.8. Tétel ([10] Theorem 2**).** *Tetszőleges T fára és tetszőleges színfüggetlen $w : E(T) \to \mathbb{N}$ súlyfüggvényre minden $\chi : L(T) \to C$ levélszínezés esetén teljesül*

$$\ell(G,\chi) = p(G,\chi).$$

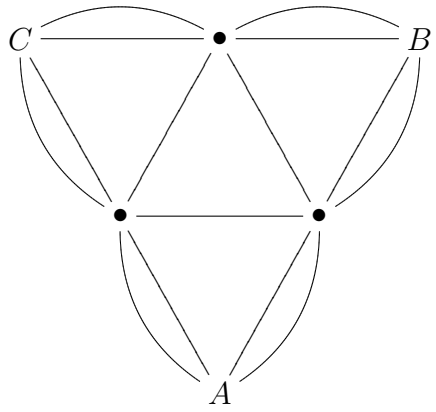
A bizonyítás itt is az útpakolás polinom időben történő, rekurzív megkonstruálásával történik.

A cikk (hasonlóan a [1] cikkhez) tartalmazza a feladat egy, a lineáris programozás nyelvén megfogalmazott variánsát, amely jelentősen különbözik a multiway cut szokásos LP megfogalmazásaitól.

Érdemes megjegyezni, hogy bár általános súlyfüggvény esetén is van polinomiális algoritmus egy optimális multiway cut megkeresésére, de itt, ellentétben a korábbi esetekkel, már nem tudtuk leírni az összes optimális multiway cut szerkezetét. Továbbá az előző minimax tétel ebben az általánosságban már is nem teljesül: ezzel a kérdéssel a Székely Lászlóval közös [2] cikkben foglalkoztunk. A cikk egy parciális színezés olyan kiterjesztéseire ajánl minimax eredményt, ahol a színezés rendelkezik egy rekurzívnak nevezett speciális tulajdonsággal.

Megjegyezzük, hogy mint azt Frank András kimutatta (lásd [13]), a fastruktúra igen hangsúlyos szerepet játszik a minimax tétel érvényességében. Már három szín mellett is lehet találni olyan "majdnem körmentes" gráfot,

1. ábra. Ellenpélda a 1.4 Tételre S-sel nem lefedett kört tartalmazó gráf esetén ($S = \{A, B, C\}, \pi_S = 8, \vec{\nu}_S = 7$)



amelyre már nem teljesül a minimax tétel. (Lásd az 1. ábrát!) Azt is érdemes megjegyezni, hogy Székely Lászlóval közösen találtunk egy olyan "jobb" alsó becslést a multiway cut problémára, amely sohasem rosszabb az eddig ismertetetteknél, és amely például a Frank féle ellenpéldában éppen kellő méretű útpakoláshoz vezet. Azonban még nem sikerült meghatározni olyan, az előzőeknél tágabb gráfosztályt, ahol az új alsó becslés mindenütt egyenlőséggel teljesülne.

2. Az evolúciós fák sztochasztikus elmélete

Ebben a fejezetben olyan problémákat tárgyalok, amelyek ugyan tisztán matematikai jellegűek, és amelyek nagy apparátust mozgatnak meg, azonban eredetük egyértelműen a biológiához köthető. A problémák háttere egy széles körben elfogadott biológiai modell, amely szerint az élővilág fejlődése, az új fajok kialakulása véletlen eseményeken alapul. A un. Kimura modell számba veszi ezen véletlen mutációk törvényszerűségeit, de nem foglalkozik azzal a kérdéssel, hogy a keletkezett egyedet mi tesz képessé a túlélésre, azaz mikor válhat egy új faj ősévé. A modell helyességének eldöntése nélkül (ez a kérdés egy matematikus számára amúgy is támadhatatlan) le kell szögezni, hogy a modellt világszerte száz és száz kutatócsoport tette vizsgálatainak alapjává.

A fejezet két alapvetően különböző megközelítést tárgyal, ezek találhatók az első két szakaszban. Az egyik egy un. karakter alapú módszer, amely minden rendelkezésre álló információt párhuzamosan használ, ezért nagy biztonsággal tudja a keresett evolúciós fát felépíteni, de eléggé lassú. A módszer lényegében két valószínűség eloszlás között fennálló Hadamard, vagy általánosabban Fourier transzformációs kapcsolatot használ fel. Ennek megfelően a neve *Hadamard konjugáció*, esetleg *Fourier párok* módszere, de *spektrál elméletnek* is nevezik. Hivatkozott cikkeim közül a [3, 4, 5, 6, 8, 11] dolgozatok foglalkoznak az említett módszerrel. Mivel a szakaszhoz tartozó cikkek lényegi részét képezték Székely László disszertációjának, amelyet a "Matematikai Tudományok Doktora" címért nyújtott be, ezért itt csak utalás szerűen térek ki a témára, főleg arra koncentrálva, milyen utóélete van ezeknek a dolgozatoknak.

A második megközelítés un. quartet alapú: ilyenkor egy evolúciós fa ismert levél-négyeseiből történik az evolúciós folyamat rekonstrukciója. Ezt a módszercsaládot általában a távolság alapú eljárások közé helyezik (bár ez nem törvényszerű): a négy levél által meghatározott részfa rekonstrukciója a levelek páronkénti (mért, számított, becsült) távolságán alapul. A [12, 14, 15, 16, 17, 18] cikkek megalkották az un. "Short quartet módszereket", közben megteremtették a különféle faépítő algoritmusok analíziséhez megfelelő környezetet. Elmondhatjuk, hogy új elméleti alapokra helyeztük a távolság alapú faépitő algoritmusokat, jelentős áttörést érve el vele úgy az algoritmusok sebességében, mint megbizhatóságában.

A két szakasz cikkeinek utóéletét legjobban a szakirodalomra gyakorolt hatásukkal lehet jellemezni. Ezt döntően a szakaszok végére hagyom. Itt csak annyit említek meg, hogy a Hadamard konjugáció alapú módszer már

megjelenése után három évvel részletes ismertetésre került egy biológusok alapképzését megcélzó tankönyvben ([SwoOls96]). Megjegyzem továbbá, hogy az evolúciós fák elméletének két, jelenleg alapvetőnek számító kézikönyve ([Fel03, SemSte03]) az itt felsoroltak közül jónéhány cikket részleteiben is ismertet. Azt is érdemes megemlíteni, hogy a kifejlesztett módszerek több kommersziális illetve szabadon hozzaférhető programcsomagban is megtalálhatók: ilyenek például a SplitsTree4, a SPECTRUM, illetve a PAUP és Molphy programcsomagok.

A fejezet utolsó szakasza ugyan nem evolúciós fák egy klasszikus értelemben vett rekonstrukciós eljárását tárgyalja, azonban mégis itt a helye. Egy 2004-es cikk alapján ([21]) egy, a supertree módszerek közé (is) besorolható eljárást ismertetek fák rekonstrukciójáról.

2.1. Hadamard konjugáció

Az 1980-as évek elején M. Kimura japán biológus egy 3-paraméteres, véletlenen alapuló mutációs modellt dolgozott ki a fajok változékonyságának megmagyarázására. Mára ez vált a biológusok által legelfogadottabb modellé. Az az alapfelvetése, hogy az élőlények átörökítő anyagában a változások teljesen véletlenszerűen, egymástól nem befolyásolva zajlanak le.

Ebben a modellben az átörökítő anyagot egy négyelemű ábécé A, G, T, C betűiből álló hosszú lineáris *szál*-ként (avagy *szó*-ként) célszerű elképzelni. A betűk négy *nuklein sav bázist* jelölnek, ezek a *Adenine* és *Guanine* (gyüjtőszóval *Purine*, ezek a két-gyűrűs bázisok) illetve a *Thymine* és *Cytosine* (gyüjtőszóval *Pyrimidine*, ezek az egy-gyűrűs bázisok). A szálaknak egyértelmű *iránya* van, amely mentén történik a tárolt információ feldolgozása. Végül alapesetben az átörökítő anyag két, egymáshoz képest *complementary, antiparallel* szálból áll. A fogalmak azt jelentik, hogy a szálak párhuzamosak de ellentétes irányúak, továbbá minden egyes, azonos pozícióban levő bázispár között kovalens foszfor kötés keletkezik. A kötések mindig az $A-T$ és $G-C$ párok között jönnek létre, azaz az egyik szálon található bázis egyértelműen meghatározza a másik szálon vele szemben található bázist. Erre utal a complementary kifejezés.

A biológusok az éppen vizsgált fajok fejlődéstörténetét a következő módon szemléltetik: Ha ismernénk a fajfejlődést leíró evolúciós fát, akkor a vizsgált fajok közös őse lenne a fa gyökere, míg a vizsgált fajokat a levelek szemléltetik, végül a leszármazás folyamán kialakult (azonban esetleg már ki is halt) "közbülső" fajokat a belső, 3-fokú elágazási pontok jelölik. Ezután minden

egyes fajt egy-egy k hosszú sorozattal jellemezhetünk, amelynek elemei az A, G, C, T betűk közül kerülnek ki. A fajok változásai pedig úgy jelentkeznek, hogy az ős és a közvetlen leszármazott fajokat (egy meghatározott élen fekvő csúcsokat) leíró k hosszú szavak bizonyos koordinátákban különböznek. (Általában, minél közelebbi rokon két faj, annál több közös elem van az őket leíró k-szavakban.)

Most a Kimura modell szerint az élek mentén lejátszódó betű-változások egymástól függetlenül, véletlenszerűen történnek. Mivel a fejlődés a közös őstől a ma élő fajok irányában történik, ezért a változásoknak egyértelmű iránya van, azonban a Kimura modell szerint egy változásnak és az ellentett változásnak ugyanannyi a valószínűsége. A modell további feltevése, hogy bár az egyes éleken a változások valószínűségei eltérőek lehetnek, azonban az ezt leíró mátrix szerkezete állandó: a mátrix sorait az őst leíró vektor adott pozíciójában található betűk indexelik, míg az oszlopokat az utód megfelelő betűi. A mátrix bejegyzései pedig azt a valószínűséget adják meg, amivel a jelzett változás bekövetkezhet. Az adott mátrix ugyan függhet az éppen jellemzett éltől, de attól nem, hogy ezen belül melyik pozícióhoz tartozik. Továbbá minden lehetséges mátrixban az egyes sorok egymás permutációi: A lehetséges változások (nincs változás, vagy a három másik betű egyike jön létre) tartozó valószínűségek négy biokémiai változást írnak le, amelyek a kiinduló betűtől függetlenül azonos valószínűséggel történhetnek meg.

Mindezen tulajdonságok alapján vezethette be Evans és Speed azt a modellt ([EvaSpe93]), ahol az egyes éleken történő változásokat ugyancsak az A, G, C, T betűkkel lehet leírni: a karakter kezdeti értéke, az élen ható változás, végül a karakter megváltozott értéke a betűkön megadott négy elemű Klein csoport hatásaként értelmezhető. Ez azt jelenti, hogyha ismerjük az őst és a leszármazottat leíró k-vektorokat, akkor meg tudjuk mondani, hogy az egyes karakterekben milyen típusú változások történtek. Másfelől ha tudjuk az ős k-vektorát, illetve az élen ható változások vektorát, akkor ki tudjuk számítani az utódot jellemző karaktereket. Érdekes megjegyezni, hogy a Klein csoport definiálta változásoknak biológiai leírását is meg lehet adni.

Ebben a modellben már könnyen megérthető a véletlen változások generálta "fejlődés". Induljunk ki a fa topológiájából, és a gyökérben található fajt jellemző k-vektorból. Ezután a véletlen fejlődés úgy történik, hogy a gyökértől elindulva és a levelek felé közeledve minden élre megadjuk az ott érvényes átmenet valószínűségek mátrixát, továbbá ennek alapján az élen minden karakterben véletlenül választunk egy átmenet típust. En-

nek segítségével ki tudjuk számolni az utód k-vektorát, továbbá, hogy mi a valószínűsége annak, hogy az ősből pont ez az utód jön létre. A teljes kiértékelés elvégzése után most meg tudjuk határozni, hogy mi a valószínűsége annak, hogy az adott topológia, gyökér színezés és átmenet mátrixok esetén éppen az adott levél konfiguráció jön létre.

Ilyenkor az éleken illetve a leveleken található színelosztások között – bizonyos ésszerű megszorítások mellett (amelyek a gyakorlati problémák esetén általában automaikusan teljesülnek) – egy Fourier inverz párkapcsolat van, amely miatt valamelyik elosztásból pontosan meghatározható a másik eloszlás. Ha az átmenet valószínűségek csak attól függnek, hogy purin-pyrimidin átmenet vagy megmaradás történik, akkor a Fourier kapcsolat egy Hadamard konjugációs kapcsolattá egyszerűsödik.

Ezek után a leveleket létrehozó lehetséges fák közül úgy lehet választani, hogy olyan fát keresünk (a fához hozzá tartozik a topológiája továbbá az előbb említett valószínűség elosztások az éleken), amely legjobban approximálja a levelekben ténylegesen megfigyelhető színelosztást. Ezen a gondolatmeneten alapul az evolúciós fák un. *spektrál elmélete.* A módszer ősét (két színre), Hendy és Penny dolgozta ki ([HenPen93] - ezt a módszert hívták eredetileg az Hadamard konjugáltak módszerének).

A módszer négy színre történő általánosítása a Székely László, Mike Steel és David Penny hármassal közös [5] cikkben kezdtük meg, illetve a Mike Steellel, Székely Lászlóval és Mike Hendyvel közös [3] cikkben fejeztük be. Szintén ebben a cikkben foglalkoztunk avval a kérdéssel, hogy a gyakorlati életben, ahol a leveleken megfigyelhető eloszlások csak bizonyos hibákkal észlelhetők, hogyan lehet egy megfelelő approximációs eljárást kifejleszteni. A kapott módszert *closest tree method*-nak nevezik. A spectrál módszert a Klein csoport helyett tetszőleges véges Abel csoportra a Székely Lászlóval és Mike Steellel közös [6] cikkben általánosítottuk. Ennek közvetlen haszna ott lehet, ha a fajokat például nem DNS-kkel, hanem protein savaikkal (amiból az emberben például 20 van) azonosítjuk. A módszernek egyébként filozófiai értelemben nagy előnye, hogy képes bizonyos esetekben kimutatni, ha az adatokra teljesen "rossz" modellt kívánunk ráhúzni, azaz popperi értelemben falszifikálható.

A módszert oktató célú írások ismertették, mint például a [SwoOls96] tankönyv vagy a [Mor96] survey cikk. Felhasználták konkrét biológiai kísérletek / megfigyelések kiértékelésére is (például a [PatWal00] cikk). Mint kiderült, hasonló módszerek ismertek voltak a quantummező elméletben (lásd például, egyebek között, a [JarBas01] vagy [AllRho06]). Érdekes az is, hogy

a módszer az egyike volt a legelsőknek, amelyet evolúciós fákról evolúciós hálózatokra általánosítottak ([Bry05]).

Az evolúciós fák rekonstrukciójához már 1987-től kezdve alkalmaztak un. *phylogenetikus invariánsok*-at. Ezek olyan függvények, amelyeket ha kiértékelünk a levelekben létező "ideális" (azaz hibamentes) adatokon, akkor az érték csak azon múlik, hogy éppen milyen topológiájú fával kötjük össze a leveleket. Invariánsok egy rendszere akkor *teljes*, ha azonosítani tudja a "valódi fát": a valódi fán minden invariáns eltűnik (a függvény értke 0), amíg minden egyéb fán legalább egy invariáns nem-zérus. A nem teljes rendszerek is alkalmassak bizonyos fák hibásságának a kimutatására. (Lásd például [Lak87] vagy [NguSpe92].)

A spektrál analízis módszerének alapján a M.A. Steel - L.A. Székely - P.L. Erdős - P. Waddell szerzőnégyes [8] cikke invariánsok (polinomok) egy teljes rendszerét határozta meg. Ezt úgy lehet alkalmazni a fák rekonstrukciójára, hogy a levelek egy lehetséges 2-partíciójára (amely a reménybeli fa egy élének elhagyásával keletkezhetett) kiértékeljük az összes invariánst. Ha mindegyik értéke 0, akkor egy létező élt találtunk meg. Egyébként az él nem eleme a fának. Az pedig közismert, hogyha egy bináris fánál ismerjük az egyes élek elhagyásával keletkező levél 2-partíciókat, akkor a fa könnyen és gyorsan rekonstruálható.

A módszert, egyéb invariáns módszerek vizsgálatán kívül (lásd például a [San93] cikket), konkrét biológiai szituációk elemzéséhez használták, például a szarvasbogarak evolúciójának során a szarvak nagyságának a hatását elemezték vele ([EmlMar05]). Sok cikk DNS sorozatok elemzésén kivül génsorozatok elemzésére is használja (pld. [AllRho04]), illetve ma már az algebrai geometria módszereit is alkalmazzák vele kapcsolatban ([EriRan04]).

2.2. A Short Quartet módszerek

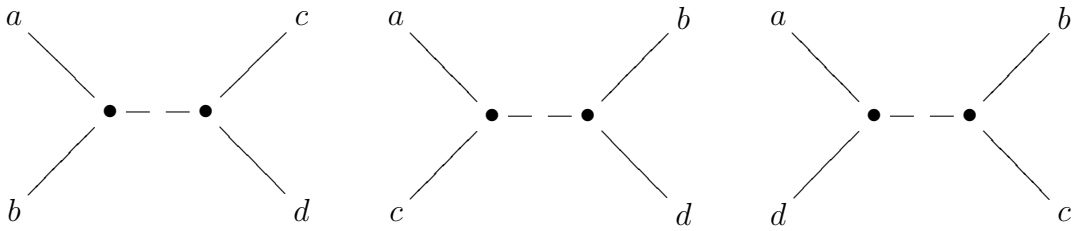
Ebben a szakaszban egy egészen más megközelítést írunk le evoluciós fák rekonstrukciójára. Jelölje $B(n)$ az n címkézett levéllel ámde címkézetlen elágazási pontokkal bíró, gyökértelen fák halmazát. (Ezeket *féligcímkézett* fáknak, avagy *X-fáknak* (angolul *X-tree*nek) is nevezik. Azért használom a szakaszban az X-fa kifejezést, hogy érzékeltessem a szélesebb kontexust.)

Legyen T egy $B(n)$-beli X-fa és legyen S a levelek egy részhalmaza. Ekkor jelölje $T_{|S}$ az S által generált részfát, míg jelölje $T^*_{|S}$ a generált bináris (topológikus) részfát (azaz minden kettő fokú belső pontot a két szomszédos éllel együtt egyetlen élbe húzunk össze). Ha adott az S levélhalmazon egy

T-vel jelölt X-fa, akkor a fa egy élének a törlése egy 2-partíciót hoz létre a leveleken, amit a továbbiakban *split*-nek nevezünk. Ha mindkét osztály legalább két levelet tartalmaz, akkor a split *nem-triviális*. Buneman régi tétele, hogy bármely féligcímkézett fát egyértelműen meghatároznak nem-triviális splitjei ([Bun71]).

Világos, hogy egy négy-levelű féligcímkézett fának (ezeket *quartet*-nek nevezzük) a három potenciális nem-triviális splitjéből pontosan egy teljesülhet egy fában: Legyen $q = \{a, b, c, d\}$ egy T-beli levél-négyes. Azt mondjuk,

2. ábra. **Splitek**: Négy pont három lehetséges splitje: $ab|cd, ac|bd, ad|bc$. Ebből egy érvényes.



hogy a $t_q = ab|cd$ egy *érvényes* (angolul *valid*) quartet split, ha ez a generált $T^*_{|q}$ bináris részfának a valódi, a fában szereplő splitje. Jelölje $Q(T) = \left\{t_q : q \in \binom{[n]}{4}\right\}$ a T X-fa összes érvényes quartet splitjét. A jól ismert, a pszichológus Colonius és Schulze nevéhez fűződő klasszikus eredmény szerint bármely T fára a $Q(T)$ halmaz egyértelműen meghatározza a T-t. Ez az eljárás, mint az könnyen látható, polinomiális időben végrehajtható.

Erre a tényre igen sokféle evolúciós fa rekonstrukciós módszert alapoztak (vagy próbáltak meg alapozni). Elvben egy ilyen úgy működhetne, hogy a módszer első fázisában valamilyen módon minden quartetre meghatározzák az érvényes splitet, majd a második fázisban ezekből felépítik a fát. (Pontosabban szólva ilyenkor a fa topológiáját lehet megkapni, de egy adott fa egy élének hosszát – azaz a változás lezajlásához elegendő időt, amely fordítottan arányos a változás valószínűségével – már nem nehéz viszonylag gyorsan meghatározni.)

Az ezen az elképzelésen alapuló egyszerű módszerek a gyakorlatban azoban meglehetősen rosszul teljesítenek. Ennek az az oka, hogy szinte sohasem sikerül minden quartetre meghatározni az érvényes spliteket, az eredmények

általában ellentmondóak. Az eljárások ennek a helyzetnek a leküzdésére sokféle stratégiát alkalmaznak, amelyek azon alapulnak, hogy valamilyen módon eldöntik, hogy a kiszámított splitek közül melyiket ismerik el érvényesnek, majd ezekből kisérlik meg helyreállítani a fát. Ezen "klasszikus" módszerek közül talán a K. Strimmer és A. von Haeseler nevéhez fűződő "quartet puzzling" eljárást használják a legtöbbet ([StrHae96]). Több hasonló módszert fejlesztettek ki, például Kearnay és kollégáinak "quartet cleaning" módszerét és annak utódait ([BerKer99]), vagy a Kanadában dolgozó magyar Csűrös Miklós nevéhez fűződő "harmonic greedy triplets" módszert (lásd a [CsuKao99] cikket).

Egyébként annak a meghatározása, hogy quartet splitek egy rendszeréhez létezik-e X-fa, amelyben ezek érvényes splitek lennének, NP-nehéz feladat. (M. Steel eredménye.)

A hibásan rekonstruált quartetek léte tehát erősen megnehezíti a quartet módszerek alkalmazását. Azonban a rosszul rekonstruált quartet splitek léte sajnos nem kellemetlen véletlen, hanem majdnem törvényszerű hiba. Mint azt nem túl bonyolult számításokkal ki lehet mutatni, a fák topológiájára és az eloszlásokra tett nagyon is ésszerű feltételek között a gyakorlati alkalmazásokban ilyen hibák majdnem biztosan előfordulnak. A jelenségnek az az oka, hogyha a quartet által meghatározott részfában (relatíve) hosszú utak vannak, akkor az út két végén levő két levél színe (karakter állapota) lényegében független egymástól (akárhány mutáció lehet közöttük).

A kutatócsoportunk által bevezetett "short quartet" módszereknek éppen az a lényege, hogy a fát viszonylag rövid quartetjeiből rekonstruáljuk, továbbá, hogy már a quartetek rekonstruálása előtt megmondjuk, melyik quartetek kerülnek felhasználásra. A csoport tagjai: Mike Steel, Székely László, Tandy Warnow és jómagam.

Először a következő problémát kell megoldanunk: tegyük fel, hogy adva van érvényes quartet splitek egy (nem teljes) rendszere. A kérdés az, hogy milyen módon és mikor lehet a rendszerből meghatározni a keresett T fát. (Vegyük észre, ez egy determinisztikus kérdés, a quartetek rekonstrukciójának esetleges hibái itt nem számítanak.)

Erre többféle módszer is ismeretes. Egy lehetséges mód az, hogy a rendelkezésre álló érvényes quartet splitek felhasználásával, az eredeti adatok további vizsgálata nélkül, meghatározzuk a többi splitet. Könnyű például belátni,

$$\text{ha } ab|cd \text{ érvényes quartet split } T\text{-ben}, \tag{3}$$

akkor $ba|cd$ és $cd|ab$ hasonlóan érvényes.

A három splitet egyébként megegyezőnek gondoljuk. Világos, ha (3) teljesül, akkor $ac|bd$ és $ad|bc$ splitek nem érvényes splitjei a T fának, ezek ilyenkor *ellentmondanak* (3)-nak.

Az előzőhöz hasonló *következtetési szabályokat (inference rule)* már eléggé sokat vizsgálták. Hasonlóan könnyen megérthető a következő következtetési szabályok érvényessége:

$$\begin{aligned}&\text{ha } ab|cd \text{ és } ac|de \text{ érvényes quartet splitek } T\text{-ben,}\\&\text{akkor szintén érvényesek az } ab|ce,\ ab|de, \text{ és } bc|de \text{ splitek;}\end{aligned} \tag{4}$$

továbbá

$$\begin{aligned}&\text{ha } ab|cd \text{ és } ab|ce \text{ érvényes quartet split } T\text{-ben,}\\&\text{akkor } ab|de \text{ is érvényes.}\end{aligned} \tag{5}$$

Ezek a szabályok *diadikus*-ak, hiszen két érvényes splitből gyártunk egy harmadikat. (Ezeket a szabályokat M.C.H. Dekker vezette be az irodalomba.) Azt mondjuk, hogy érvényes quartet splitek egy rendszere *szemi-diadikusan* meghatározza a T fát, ha a (3) és (4) szabályok rekurzív alkalmazásával előállítható a fa minden érvényes quartet splitje (és persze csak azok). Ha még a (5) szabályt is felhasználjuk akkor *diadikus* előállításról beszélünk. Maga az eljárás, amikor rekurzívan kiszámítjuk az új quartet spliteket az eredeti quartet halmaz *(szemi-)diadikus lezárása.*

A [12] preprint egyik fő eredménye a következő: jelölje $L_T(q)$ a q nevű quartet generálta $T_{|q}$ (nem feltétlenül bináris) részfában a leghosszabb, a $T^*_{|S}$ fában egy élbe összehúzódó út élszámát. Ekkor teljesül:

2.1. Tétel ([12]). *Legyen $T \in B(n)$ legalább négy levéllel. Jelölje $D(T)$ az összes olyan quartet halmazát, amelyekre $L_T(q) \leq 18 \log n$. Ekkor $D(T)$ szemi-diadikus lezárása a levélszám függvényében polinomiális időben előállítja a fát.*

Ez egy determinisztikus eredmény, amely a féligcímkézett fák definícióján kívül semmit sem használ fel, tehát független attól, hogy az evolúciónak milyen modelljét alkalmazzuk. Azonban lehetővé tette az irodalomban megtalálható első olyan evolúciós fa rekonstrukciós algoritmus megszerkesztését, amelynek teljes valószínűségi analízise elvégzésre került (mindez a purine-pyrimidine párok cseréjére vonatkozó szimmetrikus, un. Cavander-Farris

modellre történt). Az analízis lényeges pontja annak meghatározása, milyen hosszú sorozatok elégségesek a levelek jellemzésére, hogy a rekonstrukciós eljárás lényegében 1 valószínűséggel határozza meg a keresett fát. Az algoritmus elméleti jelentőségét az adja, hogy - véletlenül - ez az elégséges karakter szám nagyon közel van a szintén ebben a cikkben meghatározott információelméletileg szükséges minimális hosszhoz, ami nagy n estén durván $\log n$. Az is fontos, hogy a futásidő is polinomiális (bár nem túl jó paraméterekkel).

Érdemes még megemlíteni, hogy az információelméleti alsó korláton kívül szintén meghatározásra került az egyik népszerű rekonstrukciós eljárás, az un. maximum compatibilty módszer által megkövetelt minimális sorozat hossz, amely $O(n \log n)$. Az is érdekes továbbá, hogy a quartetek rekonstrukciójára a módszer az előző szakaszban említett invariáns módszer egy speciális változatát használja, amely szintén újszerű.

A Mike Stellel, Székely Lászlóval és Tandy Warnowval közös 1997-es [14] cikk a 2.1. Tételre talált jelentős élesítést. Egy T evolúciós fában egy él *mélysége* (*depth*) az éltől a lehető legközelebbi levélhez vezető út élszáma. A fának magának a $d(T)$ *mélysége* pedig a benne található legnagyobb él mélység. Például a "szőrős hernyó" mélysége (egy út lelógó élekkel) csak 1, míg a legnagyobb lehetséges mélység is lényegében csak $\log_2 n$ (egy teljesen kiegyensúlyozott bináris fánál).

2.2. Tétel ([14] Theorem 2**).** *Legyen T egy X-fa n levéllel és legyen*

$$D(T) = \left\{ q \in \binom{[n]}{4} : L_T(q) \leq 2d(T) + 1 \right\}$$

ahol csak olyan 4-levelű részfákat veszünk figyelembe, amelyek középső útja egyetlen élből áll. Ekkor T meghatározható a $D(T)$ szemi-diadikus lezártjából.

Ugyanezek a szerzők 1997 és 1999 között egy sorozat cikket publikáltak a Short Quartet algoritmus sémáról ([15, 16, 17, 18]). (A módszereket együttesen *Short Quartet Módszerek*nek (avagy *SQM*) nevezik.) Röviden összefoglalva a séma algoritmusai a következő módon épülnek fel:

Short Quartet algoritmusok sémája

(i) a feladat inputja quartetek egy rendszere,

(ii) amelyekből valamilyen módszerrel kiválasztjuk a rövid quarteteket,

(iii) rekonstruáljuk a kiválasztott rövid quartetek részfáit,

(iv) a rekonstruált quartetekből helyreállítjuk a fát,

(v) az eljárás közben felismerjük, ha a kiválasztott kvartet rendszer alkalmatlan a fa rekonstruálására (ellentmondó, vagy nem elégséges),

(vi) a (ii)-(v) lépéseket addig ismételjük, amíg megkapjuk a fát, avagy felismerjük, hogy nem lehetséges a rekonstrukció.

Érdemes itt kitérni a biológiai és matematikai szemléletmód különbözőségére: a szerzők, Karl Popper szellemében, a séma erősségének tekintették a falszifikálás képességét: a módszer felismerte, ha az input elégtelen vagy ellentmondó. Ugyanakkor a biológusok a rendszer hátrányának tekintették, hogy a séma nem minden esetben rekonstruál egy fát. Az ellentmondást napjainkban oldották fel, méghozzá kézenfekvő elvek szerint: E. Mossel és munkatársai ([DasHil06]) kidolgozták az SQM olyan változatait, amelyek a lehető legnagyobb, még biztonsággal rekonstruálható erdőt (azaz az "igazi fa" pontdiszjunkt részfáinak egy rendszerét) szolgáltatják.

A [16] cikk az általános módszer extended abstractjának tekinthető, rövid összefoglalóját adja. A [15] cikk a módszerek biológiai relevanciáját próbálta leírni. Az elmélet szigorú kidolgozása a [17, 18] cikkekre maradt.

A [17] cikk először is teljes általánosságban bebizonyítja az információelméleti alsó korlátot egy X-fa determinisztikus vagy véletlen módszeren alapuló rekonstrukciójához szükséges minimális sorozat-hosszra.

Másodszor bebizonyítja a 2.2. Tétel egy még erősebb változatát. Ehhez először is bevezetjük a *reprezentatív quartetek* fogalmát. Egy n levelű X-fa mind az $n-3$ belső éléhez hozzárendelünk pontosan egy reprezentatív quartetet. Ez olyan quartet, amelynek középső útja megegyzik az éllel, a négy hozzátartozó levelet pedig a következő módon határozhatjuk meg. Elhagyva az élt, továbbá közvetlen környezetét, négy darab gyökeres részfát kapunk. Minden részfában megkeressük a gyökérhez (topológiában) legközelebbi levelek közül a legkisebb címkét hordozót. Az így meghatározott négy levél alkotja a keresett reprezentatív quartetet. (Megjegyzendő, hogy minden reprezentatív quartet automatikusan rövid.) Ezután a cikk megmutatja, hogy:

2.3. Tétel ([17] Sec. 4.2**).** *A reprezentatív quartetek diadikus lezártja egyértelműen meghatározza a fát.*

(Mind látható, a megkívánt quartetek számának csökkenése maga után vonja, hogy (3), (4) és (5) következtetési szabályok mindegyikét fel kell használni.) A cikk ezután leírja az SQM egyik megvalósítását, a Dyadic Closure Tree Construction algoritmust (rövidítve DCTC algoritmust). Az algoritmus eredményeit a következő módon lehet összegezni:

2.4. Tétel ([17] Theorem 6**).** *Legyen a Q quartet splitek egy rendszere. Ekkor:*

(i) *Ha a DCTC meghatároz egy fát Q-ra, és egy másikat quartet splitek egy bővebb rendszerére is, akkor a két fa megegyezik.*

(ii) *Ha a DCTC eredménye inkonzisztens, azaz ellentmondó quartet splitek is keletkeznek, akkor hasonló történik minden bővebb quartet rendszerre is.*

(iii) *Ha a DCTC nem képes Q-ból kiszámolni a fát, akkor hasonló a helyzet bármely szűkebb quartet rendszerre is.*

(iv) *Végül ha Q ellentmondás mentes és eleme minden reprezentatív quartet, akkor a DCTC előállítja a fát.*

Megjegyzendő, hogy a cikk a DCTC algoritmusra egy $O(n^5)$ implementációt mutat be. Továbbá természetesen az is igaz, hogy a Q diadikus lezártja akkor is előállíthatja a T-t, ha nem minden reprezentatív quartet szerepel benne.

A DCTC algoritmus-magra sokféle faépítő algoritmust lehet alapítani. Ezek mindegyikének quartetek egy-egy Q halmazát kell meghatározni, amely eléggé bő ahhoz, hogy tartalmazza az összes reprezentatív quartetet, de eléggé szűk ahhoz, hogy ne legyen ellentmondó. Az Short Quartet Módszer séma alapfeltevése az, hogyha sikerül a Q meghatározásakor csupa rövid quartet felhasználni, akkor az ellentmodásmentesség automatikusan teljesül.

Természetesen pontosan a rövid quartetek kiválasztása a nehéz: az utak hosszúsága egy topológikus mennyiség, a benne foglalt élek számával azonos. A megfigyelt adatok azonban nem tartalmaznak erre direkt utalást. Egy lehetőség, ha a mért adatokra valamilyen távolság függvényt illesztünk, és ennek alapján próbáljuk meg kiválasztani a topológikusan rövid quarteteket. Nem szabad azonban elfelejteni, hogy ezek a mennyiségek matematikai értelemben nem igazi távolságok: nem csak a háromszög-egyenlőtlenséget nem teljesítik, de gyakran nem is kommutatívak. Egy másik probléma, hogy egy rövid quartethez négy végpont szükséges, és a középső élhez illeszkedő

mind négy útnak rövidnek kell lenni. Azonban mind a $\binom{n}{4}$ lehetséges négyesre ellenőrizni a hosszat nagyon lassú. Végül itt érdemes megemlíteni a módszer azon előnyét, hogy a Q-ba felveendő egyes quartet splitek megállapításához egyéb, akár kevert módszereket is lehet alkalmazni.

Egy lehetséges stratégiát a *Diadic Closure Módszer* (DCM) ír le: a DCM egy távolság-becslés alapú eljárással dönti el, hogy mely quarteteket kívánja rekonstruálni, magát a rekonstrukciót pedig a még Buneman által bevezetett un. *four point* módszerrel hajtja végre. Mint a cikk következő szakaszában található, eléggé terjedelmes valószínűségi analízis megmutatja, a paraméterek egy meglehetősen széles tartományában a DCM nagy valószínűséggel helyesen rekonstruálja a fát, és futásideje nem rosszabb, mint $O(n^5 \log n)$. Ami azonban sokkal fontosabb, a módszer viszonylag rövid, az elméleti határhoz közeli hosszúságú sorozatok ismeretét követeli meg a helyes rekonstrukcióhoz. Pontosabban:

2.5. Tétel ([17] Theorem 9**).** *Tegyük fel, hogy a Cavender-Farris modell alatt k karakter fejlődik a T evolúciós fa mentén, ahol minden e élen a változás valószínűségére teljesül $p(e) \in [f, g]$, ahol f és g az n függvényei. Ekkor a DCM módszer $1 - o(1)$ valószínűséggel rekonstruálja a T fát, amennyiben a karakterek számára teljesül a*

$$k > \frac{c \cdot \log n}{(1 - \sqrt{1-2f})^2 (1-2g)^{4depth(T)+6}} \tag{6}$$

összefüggés (ahol c valamilyen rögzített konstans).

Mint a tételből látható, a szükséges sorozat-hossz a fa mélységétől függ, amíg más ismert módszerek hatékonysága általában a fa átmérőjének a függvénye. Ezért a [17] dolgozat ezután két gyakran tekintett valószínűségi eloszlás mellett elemzi a fák mélységét és átmérőjét. A két eloszlás: az egyenletes, ahol minden fa egyformán valószínű, és a Yule-Harding féle, amelynél a "lombosabb" (ezért időben hamarabb kifejlődő) fák valószínűsége nagyobb.

A kapott eredmények alapján ezután a DCM módszer hatékonysága és érzékenysége két másik, szintén (akkor) frissen fejlesztett és közkedvelt módszer paramétereivel kerül összehasonlításra. Az egyik a *neighbor-joining* algoritmus (közkeletű rövidítéssel *NJ*), a másik pedig az Agarwala és társai által kifejlesztett 3-approximációs algoritmuson alapul, amely az L_∞ normában legközelebbi fát keresi. Ez utóbbi alapján Farach és Kannan fejlesztett ki X-fa rekonstrukciós eljárást. Mindkettőnek van worst-case analízise, amely

alapján módszereikre a szükséges sorozat hosszat a (6) formulához hasonló egyetlőtlenség becsli, de ahol a fa mélysége helyett az átmérő szerpel. Ezért a DCM sohasem rosszabb náluk, de általában lényegesen előnyösebb.

Érdemes talán megemlíteni, hogy a neighbor-joining módszer konzisztenciáját bizonyító Atteson cikk ([Att99]) intenzíven használja a [18] cikk eredményeit.

A cikksorozat utolsó cikke ([18]) először különféle távolság alapú fa-rekonstrukciós algoritmusok hatékonyságának összehasonlítására fejleszt ki egy módszert. Az ilyen módszerek általában szólva nem a levelekben lévő karakter-sorozatokkal magukkal foglalkoznak, hanem először meghatározzák az egyes levelek egymástól való "távolságát", amely a sorozatok "nem hasonlóságán" (*dissimilarity*) alapulnak: minél kevésbé hasonló két sorozat, annál nagyobb a távolságuk. (Itt megint hozzá kell azonban tenni, hogy ezek az értékek nem teljesítik a háromszög egyenlőtlenséget. Ennek leküzdésére már korán bevezettek bizonyos transzformációkat, amely segítenek a problémán. Azonban erre a tulajdonságra a tárgyalt algoritmusoknál nincs szükség.) Ez az elemzés sok elméleti munkában kerül felhasználásra – például a már említett Atteson cikk ([Att99]).

A cikk fő hozzájárulása a quartet módszerek témájához egy újonnan fejlesztett algoritmus. Ennek alapja a *Witness-Antiwitness Tree Construction* módszer. A WATC alapja az *edi-részfa* fogalma. (A megnevezés az angol *edge-deletion-induced* kifejezés rövidítése, amit itt az egyszerűség kedvéért használok.) Ha egy fából elhagyunk egy élt (de a végpontjaikat nem), akkor két gyökeres edi-részfa keletkezik. Két ilyen részfa *iker (sibling)*, ha pont diszjunktak és gyökereik távolsága a fában éppen 2 (azaz egy kettő élt tartalmazó út köti össze őket). Ha van kettő iker edi-részfa, akkor gyökereiket egy kettő hosszú úttal összekötve megint az eredeti fa egy edi-részfáját nyerjük. A WATC algoritmus a levelekből kindulva egyre nagyobb és nagyobb edi-részfákat konstruál meg. Egy adott pillanatban megkeres két edi-részfát, amelyet egy nagyobb részfává lehet egyesíteni egy új gyökér bevezetésével (a két eredeti gyökér ezen új pontnak lesznek a szomszédai).

Legyen adva egy T X-fa, továbbá quartet splitjeinek egy Q rendszere. Egy $uv|wx$ quartet split *tanúsító* (*witness*) a t_1 és t_2 részfa ikerségére, ha $u \in t_1$, $v \in t_2$, továbbá $\{w, x\} \cap (t_1 \cup t_2) = \emptyset$. Egy $pq|rs$ quartet viszont az *anti-tanúsító* (anti-witness) az ikerségükre, ha $p \in t_1$, $r \in t_2$, és $\{q, s\} \cap (t_1 \cup t_2) = \emptyset$

Azt mondjuk, hogy

- a Q rendelkezik a *tanúsító* tulajdonsággal a T fára nézve, ha bármely

két t_1 és t_2 iker edi-részfához (amennyiben a részfákon kívül még legalább két levél van T-ben) a Q-ban van tanúsító quartet split.

- a Q rendelkezik az *anti-tanúsító* tulajdonsággal a T fára nézve, ha amennyiben a Q-ban van tanúsító quartet a nem-iker t_1 és t_2 edi-részfák ikerségére, akkor anti-tanúsító quartet is található.

2.6. Tétel ([18], Subsetcions 4.4 – 4.6**).** *Ha a reprezentatív quartetek R_T halmaza része a Q-nak, akkor Q rendelkezik a T-re nézve a tanúsító tulajdonsággal. Továbbá, ha $R_T \subseteq Q \subseteq Q(T)$ (azaz a reprezentatív quartetek halmaza része az ellentmondás mentes Q-nak), továbbá t_1 és t_2 iker edi-részfák, akkor a Q-ban van legalább egy tanúsító quartet, de nincs egyetlen anti-tanúsitó quartet sem.*

Azt mondjuk továbbá, hogy quartet splitek egy Q halmaza T-*kényszerítő*, ha létezik egy olyan T X-fa, amelyre

1. $R_T \subseteq Q \subseteq Q(T)$,
2. Q rendelkezik anti-tanúsító tulajdonsággal a T-re nézve.

A WATC algoritmus ezek után képes gyorsan ($O(n^2 + |Q| \log |Q|)$ idő alatt) rekonstruálni a T féligcímkézett fát ha a Q quartet halmaz T-kényszerítő ([18]).

A cikkben ezután a *Witness-Antiwitness Method* (WAM) módszer leírása következik. ([18], Section 5.) Az algoritmus alapvető kérdése az, hogy hogyan kell kiválasztani quartetek egy megfelelő T-kényszerítő Q halmazát, ha adott a levelek páronkénti távolsága. A módszer többféle keresési stratégiát vezet be, amelyek függnek nemcsak az elvárt gyorsaságtól, hanem a rendelkezésre álló sorozat-hosszaktól is.

Az algoritmus valószínűségi elemzése azt mutatja, hogy a WAM sikeresen képes rekonstruálni a fát a DCM eljárásával lényegében megegyező paraméter tartományban, méghozzá lényegesen gyorsabban, mint a DCM. Az is lényeges, hogy eközben a szükséges sorozat-hossz csak kicsit múlja felül a DCM-nél szükségeset.

Érdemes még azt is megjegyezni, hogy bár az elemzéseknél feltettük, hogy minden levél azonos hosszúságú karakter sorozattal van jellemezve, azonban az algoritmusok futtatásához ez egyáltalán nem kötelező. Ennek az az oka, hogy a quartet splitek távolság-adatok helyett egyéb információk alapján is

kiszámíthatók: bármilyen más módszer elfogadható a splitek számítására, feltéve, hogy megbízható eredményeket adnak.

Ennek legfőbb jelentősége az, hogy egészen nagy adathalmazok kezelésére is alkalmasak lehetnek ezek a módszerek. Ugyanis (mint már említettük) a karakter sorozat alapú módszerek nagy adathalmazon való alkalmazhatóságának elvi határt szab, hogy nagyon divergens adatok (azaz nagyon sokféle faj együttes előfordulása) esetén egyszerűen nem létezhet elegendően hosszú, közös jellemzőket leíró sorozat. (Primitív példaként, ha például egyszerre vizsgálunk gerinces és gerinctelen állatokat, akkor persze nem állnak rendelkezésre mindkét típusra a gerinccel kapcsolatos karakterek.) Mindkét módszerünk megkerüli a problémát, hiszen lehetséges, hogy eltérő négyesekre eltérő módszereket alkalmazunk a quartet splitek meghatározására. Ezekre az esetekre azonban természetesen nem vonatkoznak az említett hatékonyság vizsgálatok.

Az SQM módszerek eddig jelentős hatást mutattak az evolúciós fák rekonstrukciójának kutatásában. Az egyik legelső példa erre a Disk Covering Method (Huson - Nettles - Parida - Warnow - Yooseph), [HusNet98]) kifejlesztése, amely módszer az SQM alapján egyéb ismert módszerek heurisztikus felgyorsitását igéri. Az E. Mossel vezette Berkeley-beli kutatócsoport egy sorozat cikkben ([DasMos06, Mos03, Mos04, MosRoc05]) jelentősen kiterjesztette az SQM-ben kifejlesztett elveket. Sok egyéb elméleti cikk is visszanyúlt ezekhez az eredményekhez (például [ChoTul05]). Végül három Science cikk is feldolgozza őket ([DriAne04], [MosVig05, MosVig06]).

2.3. X-fák és súlyozott quartetek

A fejezet utolsó szakaszában egy Andreas Dress-szel közös eredményt ismertetek ([21]).

Emlékeztetőül, a címben szereplő *X-fa (X-tree)* az evolúciós fák egy másik elnevezése, amit nem-biológusok használnak. Azért használom itt én is ezt az elnevezést, mert a módszer nem törődik avval, vajon a bemenő adatok valamilyen biológiai vizsgálatból jöttek-e. Az X-fa, értelemszerűen, egy (esetleg gyökeres) bináris fa, ahol az elágazási pontok címkézetlenek, míg a levelek egy X halmazból kapnak egy-egy értelműen címkéket.

Legyen X egy véges halmaz és jelölje $\mathcal{S}_{2|2}(X)$ az X össszes négyeseiből megalkotható 2-2 splitet, azaz

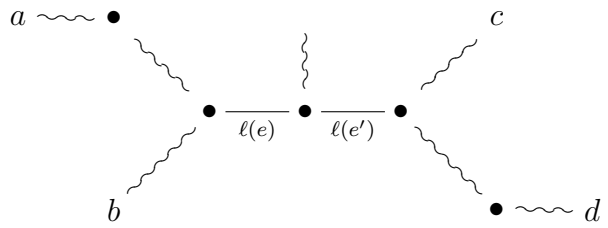
$$\mathcal{S}_{2|2}(X) \;:=\; \Big\{\Big\{\{a,b\},\{c,d\}\Big\}\Big|$$

$$\{a,b\},\{c,d\} \in \binom{X}{2};\ \{a,b\} \cap \{c,d\} = \emptyset \Bigg\},$$

Jelölje $E_1 = E_1(T)$ a T fa összes belső élét, legyen továbbá $\ell : E_1 \to \mathbb{R}_{>0}$ egy tetszőleges, de szigorúan pozitív, valós *hossz-függvény.* Minket az a $W = W_{T,\ell}$ függvény érdekel, amelyet a következő módon definiálunk $\mathcal{S}_{2|2}(X)$-en:

$$W : \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0} : ab|cd \mapsto \sum_{e \in E(ab|cd)} \ell(e) \tag{7}$$

ahol az összegzés a $E(ab|cd)$ halmazra történik, amely az összes olyan $e \in E$ élt tartalmazza, amely a T fában szeparálja az a, b leveleket a c, d levelektől. A W függvény nyilván a $T|_{\{abcd\}}$ részfa "középső részének" hosszát méri, amennyiben a $ab|cd$ egy érvényes split, egyébként pedig nulla az értéke.



Most könnyen ellenőrizhető, hogy egy teteszőleges X-fára és tetszőleges hossz-függvényre teljesülnek a következő tulajdonságok:

(F1) Bármely X-beli, 4-elemű $\{a,b,c,d\}$ részhalmaz esetén a $W(ab|cd), W(ac|bd)$ és $W(ad|cb)$ számok közül legalább kettő nulla.

(F2) Ha a T fa bináris, akkor bármely $\{a,b,c,d\} \in \binom{X}{4}$ négyes esetén

$$W(ab|cd) + W(ac|bd) + W(ad|cb) > 0 \tag{8}$$

teljesül.

(F3) Legyen $a,b,c,d,x \in X$ ahol $|\{a,b,c,x\}| = |\{b,c,d,x\}| = 4$ és

$$W(ab|xc), W(bx|cd) > 0,$$

akkor $|\{a,b,c,d,x\}| = 5$ és

$$W(ab|xc) + W(bx|cd) = W(ab|cd). \tag{9}$$

(F4) Bármely 5-elemű X-beli $\{a, b, u, v, w\}$ halmazra teljesül

$$W(ab|uw) \geq \min\Big(W(ab|uv), W(ab|vw)\Big). \tag{10}$$

Ezek után az idézett dolgozat fő eredménye a következő:

2.7. Tétel ([21] Theorem 1.1**).** *Egy*

$$W : \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$$

leképezés akkor és csakis akkor áll elő egy megfelelő T bináris fa, X levél címke halmaz és ℓ hossz-függvény esetén $W_{T,\ell}$ formában, amennyiben a W függvény kielégíti az (F1) - (F4) *feltételeket. Ilyenkor a W függvény illetve a hossz-függvénnyel ellátott bináris fa közötti megfelelés egy kanonikus leképezés erejéig egyértelmű.*

A tétel egyfelől a hossz-függvények axiomatizálásának tekinthető: egy quarteteken megadott függvény akkor és csakis akkor lehet egy létező X-fa hossz-függvénye, ha teljesíti a feltételeket. Másfelől a tétel bizonyítása egyben egy fa rekonstrukciós eljárást is nyújt ezekből az adatokból, amely a supertree módszerek közé sorolható (lásd például [Wil04]).

3. Szavak rekonstrukciója - DNS kódok

A szavak kombinatorikája (combinatorics on words) széles körben vizsgált, jól megalapozott területe a matematikának. Gyökerei mélyen vannak a csoport- illetve valószínűségelméletben, és sok alkalmazást talált az automaták matematikai elméletében vagy a számítógéptudományban. A vizsgált objektum általában egy véges $\Gamma = \{1, 2, \ldots, k\}$ ábécén értelmezett összes véges *szó* (avagy *sorozat*) Γ^* összessége alkotta végtelen poset, amelyet a *részsorozatnak lenni* reláció rendez el. (Ha $v_1...v_k$ és $w_1....w_\ell \in \Gamma^*$ akkor $v < w$ akkor és csakis akkor teljesül, ha $k < \ell$ és $\exists \phi : [k] \to [\ell]$ szigorún monoton növő leképezés, hogy $\forall i \in [k] : v_i = w_{\phi(i)}$, ahol, a szokott módon, $[k] = \{1, ..., k\}$.) A témáról jó bevezető az M. Lothaire álnéven publikáló francia matematikus csoport által megjelentetett [Lot97] könyv.

Ugyanezen objektumok fontos szerepet játszanak a molekuláris biológia alapvető problémáiban is. Ilyenkor a vizsgálandó rendszert leíró biológiai sorozatok a négy nukleotidát (A, C, G, T) tartalmazhatják. Ha DNS helyett RNS sorozatokat vizsgálunk, akkor a T (azaz tymine) helyett U (azaz uracyl) szerepel a sorozatokban. A sorozatok (vagy szavak) vehetik betűiket az aminosavakból is (az emberi szervezetben ebből húsz féle létezik, de az összes élőlényben sem ismeretes 26-nál több). Továbbá tekinthetjük a kromoszómákon előforduló géneket is, ahol a valódi biológiai sorozatokban az egyes gének egynél nagyobb multiplicitással és kétféle irányítással is szerepelhetnek (emlékeztetőül: a DNS szálaknak jól definiált iránya van). Ezeknél a sorozatoknál különféle véges optimalizálási számításokat kell elvégezni. Ezekkel a feladatokkal a *string* (*fűzér*) algoritmusok tudománya foglalkozik. Ebbe a témába talán Dan Gusfield könyve ([Gus97]) a legjobb bevezető.

A fejezet első szakaszában egy tisztán számítógéptudományi problémát vizsgálok meg röviden egy A. Apostolicoval és M. Lewenstein-nel közös cikk alapján ([25]). A következő szakaszokban egy véges ábécé feletti véges szó poset tulajdonságait tanulmányozzuk: előbb a hagyományos környezetben, majd a biológiában hasznos "fordított komplemens" rendezésben (a [20, 23, 26] dolgozatokat alapján). Végül néhány gondolatot írok le DNS kódokkal kapcsolatban ([22]).

3.1. Hibákat is megengedő paraméteres párositások

Ebben a szakaszban a string elmélet egyik alapvető problémájának egy általánosítását tárgyalom a [25] cikk alapján. (A cikk immár kettő éve van nyomdai

szakaszban, várhatóan 2006-ban megjelenik.) A különféle string keresések a számítógépes eljárások egyfajta alapvető "primitivjei": olyan építőelemek, amelyeket a legkülönfélébb eljárásokban használnak. A szokásos megfogalmazásánál adott egy (általában hosszú) *szöveg (text)*, és egy (általában sokkal rövidebb) *minta (pattern)*, ahol a minta összes szövegbeli előfordulását kell megtalálni. Ezt hívják a minta *párosítás*ának. Az alapprobléma sokféle változata ismert: megengedhetünk például korlátos számú hibát a minta előfordulásában, vagy törléseket illetve beszúrásokat is. A paraméteres változatban a szöveg és a minta ábécéje különbözhet egymástól, és akkor gondoljuk, hogy egy adott pozicióban a minta megjelenik a szövegben, hogyha létezik a két ábécé között olyan injektív leképezés, ami teljes aznosságot garantál. A probléma a software engeneeringben, programok tömörítésénél merült fel.

A *közelítő* (*hibákat megengedő*) paraméteres párosítás a következő feladatot jelenti: legyen $t = t_1t_2...t_n$ egy (hosszú) szöveg és legyen $p = p_1p_2...p_m$ egy (rövidebb) minta, amelyek az (esetleg) eltérő Σ_t és Σ_p ábécé fölöttiek. Ezután mindegyik i szöveg-pozicióhoz keressük azt a $\pi_i : \Sigma_p \to \Sigma_t$ injekciót, amely maximalizálja a megegyezések számát a $\pi_i(p)$ leképzett minta és a $t_it_{i+1}...t_{i+m-1}$ szövegdarab között ($i = 1, 2, ...n - m + 1$).

A probléma általános esete könnyen megoldható $O(nm(\sqrt{m} + \log n))$ lépésben, ha a kérdést a szöveg minden poziciójában visszavezetjük páros gráfok maximális súlyú párosításaira (ez már 1974-ben is ismert volt).

A [25] cikk azt az esetet vizsgálja, amikor mind a szöveg, mind a minta futamokkal van kódolva: megadjuk az első pozicióban levő betű megszakítás nélküli, (maximális számú) egymást követő előfordulásainak számát, majd megadjuk a rákövetkező betűt, és annak a multiplicitását, stb. Jelölje r_t és r_p a szövegben illetve a mintában jelenlevő futamok számát.

A dolgozat egy $O(r_p \times r_t)$ idő komplexitású algoritmust fejleszt ki arra az esetre, amikor legalább az egyik ábécé bináris. A futásidőt terheli még egy (szöveghosszban) lineáris előkészítő fázis, továbbá egy logaritmikus szervezési overhead.

3.2. Szavak rekonstrukciója - klasszikus eset

A Sziklai Péterrel és David Torney-val közös [20] cikk a véges Γ ábécéből vett szavak alkotta véges posetekkel foglalkozik: legyen $\mathcal{P}^{(n)}$ az ábécé betűiből vett összes, legfeljebb n hosszú sorozat részben rendezett halmaza. A kapott posetben a szavak hossza egy alkalmas rang függvényt határoz meg, ezért a

$\mathcal{P}^{(n)}$ poset szintezett. Jelölje $\mathcal{P}_i^{(n)}$ az i-edik szintet, amely az összes i hosszú részsorozatból áll ($0 \leq i \leq n$).

Míg a végtelen változat napjainkban rengeteget vizsgált objektum, addig a véges változat szinte semmilyen figyelmet sem kapott. Jelentőségét többek között az adja, hogy a DNS vizsgálatokban használt *törlés - beszúrás* (*delition-insertion*) metrikán (avagy Levenshtein távolságon) alapuló hibajavító kódok tanulmányozásának természetes közege lehet. Ezen szavak kombinatórikájának legfontosabb kutatója maga Vladimir Levenshtein (például [Lev92, Lev01a, Lev01b]). Egy másik fontos, korai eredmény P.J. Chase nevéhez fűződik: ő tanulmányozta egy sorozat részsorozatai számának eloszlását. Legyen S egy adott sorozat, jelölje S_i az i hosszú részsorozatok halmazát, még $|S_i|$ azok számát.

Tétel. [P.J. Chase ([Cha76])] *Az* $|S_i|$, $(0 \leq i \leq n)$ *számok egyszerrre érik el maximumukat, méghozzá pontosan akkor, amikor az* S *szó az abécé egy ismétléses permutációja, azaz egy* $(w_1 \dots w_k) \dots (w_1 \dots w_k) w_1 \dots w_\ell$ *formájú sorozat, ahol* $\ell \equiv n \pmod{k}$ *és* $w_1 \dots w_k$ *a* Γ *egy rögzitett permutációja — vagy pedig az előző sorozat fordítottja.*

A továbbiakban jelölje $\mathcal{B}_{k,n}$ a Chase Tételben leírt, maximalitást biztosító elem által generált $\mathcal{P}^{(n)}$-beli ideált, mint posetet.

3.2.1. Automorfizmusok

A $\mathcal{B}_{k,n}$ posetet G. Burosch és társai sokat vizsgálták ([BurFra90, BurGro96]). Az első cikk fő eredményeként meghatározták a $k = 2$ esetre kapott poset automorfizmus csoportját, amelyről kiderült, hogy az feltűnően "szegényes". A szerzők a $\mathcal{B}_{k,n}$ posetet először egy megfelelően választott Boole hálóba ágyazták be és annak tulajdonságait használták fel a bizonyítás során. A második cikkben, hasonló eszközökkel, a kérdést az általános ábécé esetére oldották meg.

A [20] cikkben kidolgozott módszer egyszerű bizonyítást szolgáltat Buroschék első cikkének eredményeire, miközben leírja a $\mathcal{P}^{(n)}$ poset automorfizmus csoportját is.

Jelölje $\mathrm{Aut}(\mathcal{P})$ a $\mathcal{P}$ poset automorfizmus csoportját. Nyilvánvaló, hogy a Γ abécé bármely π permutációja indukálja a $\mathcal{P}^{(n)}$ egy σ_π automorfizmusát a $\sigma_\pi(w_1 w_2 \dots w_t) = \pi(w_1)\pi(w_2) \dots \pi(w_t)$ jelölés mellett. Jelölje Sym_k az $\mathrm{Aut}(\mathcal{P}^{(n)})$ csoport σ_π automorfizmusok által generált részcsoportját. Legyen továbbá ρ azt a műveletet, amely bármely sorozatban megfordítja az elemek

sorrendjét (például $\rho(abcd) = dcba$). Ekkor ρ maga is automorfizmus, és $\rho^{-1} = \rho$. Jelölje Z_2 a $\mathrm{Aut}(\mathcal{P}^{(n)})$ csoport ρ által generált részcsoportját. Azt is könnyű látni, hogy ρ bármely másik automorfizmussal is felcserélhető.

Az $n = 2$ esetben bármely (rendezetlen) $\{a, b\} \subset \Gamma$ párra legyen ϱ_{ab} az a leképezés $\mathcal{P}^{(2)}$-n amely felcseréli ennek (és csak ennek) a két betűnek a sorrendjét, valahányszor együtt jelentkeznek egy 2-sorozatban. Ilyen leképezésből éppen $\binom{k}{2}$ van, bármely különböző (rendezetlen) $\{a, b\}$ és $\{c, d\}$ párra ezek az automorfizmusok különböznek és felcserélhetők (hiszen más párokon hatnak). Ezért ezek a ϱ leképezések együtt az identitással az $\mathrm{Aut}(\mathcal{P}^2)$ csoport egy részcsoportját képezik, amelyet $Z_2^{\binom{k}{2}}$-vel jelölünk. A rész főeredményét ezek után úgy lehet megfogalmazni, hogy a $\mathcal{P}^{(n)}$ csoport bármely automorfizmusát a Sym_k részcsoport és vagy a Z_2 vagy a $Z_2^{\binom{k}{2}}$ részcsoportok egy-egy elemének szorzataként lehet előállítani.

3.1. Tétel. (i) *Ha* $n > 2$, *akkor* $\mathrm{Aut}(\mathcal{P}^{(n)}) = Sym_k \otimes Z_2$*;*
(ii) *ha* $n = 2$, *akkor* $\mathrm{Aut}(\mathcal{P}^{(n)}) = Sym_k \otimes Z_2^{\binom{k}{2}}$.

Burosch első (bináris) cikkének eredményei most könnyen kijönnek a 3.1. Tétel bizonyítására használt gondolatmenetből. A bizonyítás továbbfejleszthető az általános ábécé esetére is: Ligeti Péter és Sziklai Péter ([LigSzi05]) ilyen módon új bizonyítást talált a [BurGro96] cikk fő tételre is.

3.2.2. Extremális kombinatorikai tulajdonságok

Most rátérünk a $\mathcal{P}^{(n)}$ poset legalapvetőbb kombinatorikai tulajdonságainak a vizsgálatára. Emlékeztetőül: posetünk szintezett, és egy sorozat rangja éppen a hossza, így $\mathrm{rang}(\mathcal{P}^{(n)}) = n$. Legyen $\mathcal{P}$ egy tetszőleges szintezett poset 0 minimális ranggal, és jelölje A az ℓ-rangú elemek egy részhalmazát. Ekkor $\Delta_i A$ jelöli ($0 \leq i < \ell$ esetén) az i-edik *árnyékát* az A-nak, míg $\nabla^i A$ jelöli ($\ell < i \leq \mathrm{rang}(\mathcal{P})$ esetén) a i-edik *felső árnyékát.*

Először is vegyük észre, hogy a $\mathcal{P}^{(n)}$ poset adott rangú elemeinek adott (i-edik) árnyékai eltérő számosságúak lehetnek. Ugyanakkor, mint kiderült, bármely két azonos hosszúságú sorozat felső j-árnyéka azonos elemszámú.

3.2. Tétel. *Legyen* ξ *egy rögzített sorozat és legyen* j *olyan egész, hogy* $|\xi| \leq j \leq n$. *Ekkor azon* j*-sorozatok száma, amelyek* ξ*-t részsorozatként*

tartalmazzák a következő:

$$N(j,\xi;k)=\sum_{i=0}^{j-|\xi|}\binom{j}{i}(k-1)^i.$$

Ezzel a tétellel egyébként új bizonyítást adtunk Levenshtein egy ismert eredményére is ([Lev92]).

Mint tudjuk, bármely posetben a BLYM egyenlőtlenségből következik a Sperner tétel. A $\mathcal{P}^{(n)}$ részbenrendezett halmaz pedig kielégíti a BLYM tulajdonságot, valamint a BLYM könnyű következménye a normalizált párosítási tulajdonságnak (normalized matching property):

3.3. Tétel. *A normalizált matching tulajdonság teljesül a $\mathcal{P}^{(n)}$ posetre, mert az i tetszőleges egész értékére és az $A\subseteq\mathcal{P}_i^{(n)}$ részhalmaz valamennyi választására:*

$$k|A|\leq|\nabla A|.$$

Az állítás egyébként a 3.2. Tétel következménye.

3.2.3. Szavak rekonstrukciója lineáris időben

Ebben a részben az Andreas Dressel közös [23] cikk alapján a véges Γ ábécé feletti n-hosszú szavak részszavaiból lineáris időben történő rekonstrukcióját tárgyalom.

Simon Imre 1975-ben válaszolta meg az általa és M. Schützenberger által még 1966 körül feltett kérdést: legyen Γ egy véges ábécé és legyen w egy n-betűt tartalmazó szó Γ felett. Tekintsük a szó összes, legfeljebb m hosszúságú részszavának $S(w,m)$ halmazát (tehát a részszavak frekvenciája nem ismert). A kérdés az, hogy az $S(w,m)$ mikor határozza meg egyértelműen a w-t, azaz milyen m-k mellett lehetséges, hogy két azonos hosszú, de eltérő w és w' szavakra megegyeznek a megfelelő részszavakból álló halmazok.

Tartalmazzon az ábécé legalább két betűt és legyen $w=ababa...ba$ míg $w'=babab...ab$. Ha mindkét szó $2m+1$ hosszú, akkor könnyen látható, hogy köztük nem tesznek különbséget a legfeljebb m hosszú részszavak halamzai. Ugyanakkor teljesül:

Tétel. [Simon (1975)] *A véges Γ ábécé felett minden $2m+1$ hosszú szót egyértelműen meghatároz legfeljebb $m+1$ hosszú részszavainak halmaza.*

A tétel legszebb bizonyítása Jacques Sakarovitch és Simon Imre nevéhez fűződik és a [Lot97] könyv 119-120. oldalán található. Itt érdemes megjegyezni, ha a részszavak halmazán kívül minden egyes részszó multiplicitását is ismerjük, akkor minden szót egyértelműen meghatároz a legfeljebb $\sim 7\sqrt{n}$ hosszú részszavainak kollekciója.

Az ismert megközelítések csupán egzisztencia bizonyítást adtak a Simon tételére, azonban nem vizsgálták a rekonstrukciót ténylegesen végrehajtó algoritmust. Ezt a munkát a [23] cikkben végeztem el, Andreas Dress-szel közösen. Az eredmény kimondásához szükség van néhány további jelölésre. Jelölje $\|w\|$ a (rész)szó hosszát, $\|w\|_a$ pedig a szóban szereplő a betűk száma, végül legyen $\binom{w}{m}$ a w szó összes m-hosszú részszavának a halmaza. A következő típusú kérdéseket teszük fel:

(i) Mennyi $\|w : m\|_a := \max\left(\|v\|_a : v \in \binom{w}{m}\right)$ azaz az m-hosszú részszavakban fellelhető a-betük maximális száma?

(ii) Mennyi $\overline{j_a}(w|m|k) := \max\left(\min\left(v^{-1}(a)\right) : v \in \binom{w}{m}, \|v\|_a \geq k\right)$ azaz mi a maximuma a legalább k darab a betűt tartalmazó m-hosszú részszavakban szereplő legelső a betű pozíciójának.

(iii) Mennyi $\underline{j_a}(w|m|k) := \min\left(\max\left(v^{-1}(a)\right) : v \in \binom{w}{m}, \|v\|_a \geq k\right)$ azaz mi a minimuma a legalább k darab a betűt tartalmazó m-hosszú részszavakban szereplő legutolsó a betű pozíciójának.

Ezután a cikk fő eredménye a következő:

3.4. Tétel ([23]). *Adott a legalább kételemű Γ ábécé, továbbá az n és m természetes számok, ahol $2m > n$. Ekkor bármely $w \in \Gamma^{[n]}$ szó rekonstruálható $|\Gamma|$ darab* (i)*-es típusú, továbbá $\lfloor n(1 - \frac{1}{|\Gamma|})\rfloor$ darab* (ii)*-es és ugyanannyi* (iii)*-as típusú kérdéssel.*

3.3. Szavak rekonstrukciója - fordított komplemens eset

Ebben a szakaszban a [26] cikk eredményeit ismertetem. Először röviden összefoglalom a genetikai anyagról szükséges ismereteket. A biológiai átörökítő anyagot hordozó DNS sorozatok a négyelemű $\Gamma = \{A, G, C, T\}$ ábécé elemeit használják. A DNS tipikusan kettős spirál alakban található, ahol a két szál egymással ellentétes irányban fut (az átörökítő anyagot feldolgozó enzimek

felismerik a szálak irányát), ahol az egyik szál A-ja mindig a másik szál egy T-jével van szemben, és hasonló kapcsolat van a C ás G betűk között.

Ennek a helyzetnek a modellezéséhez legyen $\Gamma = \{a, \bar{a}; b, \bar{b}\}$ ahol a betűk un. *komplemens párokban* vannak. Definiáljuk a következő műveleteket: $\bar{\bar{a}} = a$, $\bar{\bar{b}} = b$ továbbá valamely $w = w_1 w_2 ... w_t$ szóra legyen $\widetilde{w} = \overline{w_t}\,\overline{w_{t-1}}\ ...\ \overline{w_1}$, amelyet az eredeti szó *fordított* (*reverse*) *komplemens*ének nevezünk. Könnyen látható, hogy $\widetilde{(\widetilde{w})} = w$. Ezután **minden szót azonosítunk a fordított komplemensével.** Ezek után a fordított komplemens rendezésben $w \prec v$ (azaz az első *megelőzi* a másodikat) akkor és csakis akkor teljesül, ha w részszava v-nek vagy részszava $\widetilde{v}$. Jelölje most $S(m, w)$ mindazon legfeljebb m hosszú v szavakat, amelyek megelőzik w-t (azaz vagy w vagy $\widetilde{w}$ szavak részszavai). A Simon Imre tételének megfelelő kérdés az, hogy milyen hosszű w szavakat lehet biztosan rekonstruálni az $S(m, w)$ halmazból. (Itt is fel lehet tenni a multiplicitásos kérdést, de erről semmi sem ismert.)

Tekintsük először a következő szavakat:

$$\mathcal{F}' = \bar{a}^{2k+\varepsilon}\, a^k \quad \text{és} \quad \mathcal{G}' = \bar{a}^{2k+\varepsilon-1}\, a^{k+1},$$

ahol $\varepsilon \in \{0, 1, 2\}$ és $k \geq 1$ továbbá $(k, \varepsilon) \neq (1, 0)$. Ekkor mindkét szó hossza $3k + \varepsilon$. Egyfelöl a $\mathcal{F}'$ szó $\bar{a}^{2k+\varepsilon}$ részszava teljesíti $\bar{a}^{2k+\varepsilon} \not\prec \mathcal{G}'$ összefüggést. Másfelől könnyű ellenőrizni, hogy

$$S(2k + \varepsilon - 1, \mathcal{F}') = S(2k + \varepsilon - 1, \mathcal{G}').$$

A cikk egyik fő eredménye a következő állítás:

3.5. Tétel ([26] Theorem 2.1**).** *Minden legfeljebb* $3m-1$ *hosszú* $w \in \{a, \bar{a}\}^*$ *szót egyértelműen meghatároz a hossza, továbbá részszavainak* $S(2m, w)$ *halmaza.*

A következő példa azt illusztrálja, hogyha szavunk legalább kétféle komplemens párból tartalmaz betüket, akkor kicsit "könnyebb" a rekonstruálása. Tekintsük a következő szavakat:

$$\mathcal{F} = \bar{a}^{2k+\varepsilon}\, \bar{b}\, b\, a^k \quad \text{és} \quad \mathcal{G} = \bar{a}^{2k+\varepsilon-1}\, \bar{b}\, b\, a^{k+1},$$

ahol $\varepsilon \in \{0, 1, 2\}$ és $k \geq 1$ továbbá $(k, \varepsilon) \neq (1, 0)$. Mindkét szó hossza $3k + 2 + \varepsilon$. Egyfelöl a $\mathcal{F}$ szó $\bar{a}^{2k+\varepsilon}$ részszava teljesíti $\bar{a}^{2k+\varepsilon} \not\prec \mathcal{G}$ összefüggést. Másfelől könnyű ellenőrizni, hogy

$$S(2k + \varepsilon - 1, \mathcal{F}) = S(2k + \varepsilon - 1, \mathcal{G}).$$

A cikk másik fő eredménye a következő állítás:

3.6. Tétel ([26] Theorem 2.2**).** *Minden legfeljebb* $3m + 1$ *hosszú* ($m > 1$) *szót, amely tartalmaz betűt mind az* (a *vagy* $\bar{a}$) *mind a* (b *vagy* $\bar{b}$) *párból, egyértelműen meghatároz a hossza, továbbá részszavainak* $S(2m, w)$ *halmaza.*

Az eredmények sorát a következő észrevétel teszi teljessé:

3.7. Tétel ([26] Theorem 3.5**).** *A 3.6. Tétel akkor is igaz marad, ha a* w *szó* $k \geq 2$ *különféle komplemens párból tartalmaz betűket.*

Talán érdemes megjegyezni, hogy a bizonyításokban a nehézséget mindenütt az jelenti, hogy bár sok (megelőző) részszó van jelen, nem tudjuk róluk, hogy a szónak, vagy annak fordított komplemensének a részszavai-e. Ez ad magyarázatot arra is, miért kell ennyivel hosszabb részszavakat ismernünk a fordított komplemens esetben. Azt is érdemes hozzátenni, hogy ebben az esetben még nem ismeretes a rekonstrukció komplexitása.

3.4. DNS kódok

Az előző szakaszban leírt részbenrendezés a szokásos Levenshtein (vagy delition - insertition) metrikához hasonló távolság fogalmat eredményez. Itt is lehet ennek megfelelően hibajavító kódokat keresni. Ezeknek már a Human Genome program idején nagy gyakorlati hasznunk volt, és megkonstruálásuk kézzel, heurisztikus alapon történt. A sokszerzős [22] cikk ennek a problémának próbált elméleti megalapozása lenni. Fő célja a fogalmak és feladatok rögzítése volt. A téma meglepően népszerű, a cikk megjelenése óta eltelt szűk egy évben már jónéhány hivatkozás történt rá, a legutolsók egyike [MilKas05].

Irodalomjegyzék

A dolgozatban érintett témákban megjelent cikkek

Az Értkezéshez csatolt cikkek az alábbi listában félkövéren vannak szedve.

[1] P.L. Erdős - L. A. Székely: Evolutionary trees: an integer multicommodity max-flow – min-cut theorem, *Advances in Appl. Math* **13** (1992) 375-389.

[2] P.L. Erdős - L.A. Székely: Algorithms and min-max theorems for certain multiway cuts, *Integer Programming and Combinatorial Optimization* (Proc. of a Conf. held at Carnegie Mellon University, May 25-27, 1992, by the Math. Programming Society, ed. by E. Balas, G. Cornuèjols, R. Kannan) 334-345.

[3] M.A. Steel - M.D. Hendy - L.A. Székely - P.L. Erdős : Spectral analysis and a closest tree method for genetic sequences, *Appl. Math. Letters* **5** (1992), 63-67.

[4] L.A. Székely - P.L. Erdős - M.A. Steel: The combinatorics of evolutionary trees–a survey, *Séminaire Lotharingien de Combinatoire, (Saint-Nabor, 1992)*, D. Foata, éd, Publ. Inst. Rech. Math. Av. **498** (1992), 129–143.

[5] L.A. Székely - P.L. Erdős - M.A. Steel - D. Penny: A Fourier inversion formula for evolutionary trees, *Appl. Math. Letters* **6** (1993), 13-17.

[6] L.A. Székely - M. Steel - P.L. Erdős: Fourier calculus on evolutionary trees, *Advances in Appl. Math* **14 (1993), 200-216.**

[7] P.L. Erdős - L. A. Székely: Counting bichromatic evolutionary trees, *Discrete Applied Mathematics* **47 (1993), 1-8.**

[8] M.A. Steel - L.A. Székely - P.L. Erdős - P. Waddell: A complete family of phylogenetic invariants for any number of taxa, *NZ Journal of Botany*, **31** (1993), 289-296.

[9] P.L. Erdős : A new bijection on rooted forests, *Discrete Mathematics* **111** (1993), 179-188.

[10] **P.L. Erdős - L. A. Székely: On weighted multiway cuts in trees,** *Mathematical Programming* **65 (1994), 93-105.**

[11] L.A. Székely - P.L. Erdős - M.A. Steel: The combinatorics of reconstructing evolutionary trees, *J. Comb. Math. Comb. Computing* **15** (1994), 241-254.

[12] M.A. Steel - L.A. Székely - P.L. Erdős: The number of nucleotide sites needed to accurately reconstruct large evolutionary trees, *DIMACS, Rutgers University, New Brunswick, New Jersey, USA* 1996.DIMACS Technical Reports 96-19

[13] **P.L. Erdős - A. Frank - L.A. Székely: Minimum multiway cuts in trees,** *Discrete Appl. Math.* **87 (1998), 67–75.**

[14] **P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule,** *Computers and Artificial Intelligence* **16 (1997), 217-227.**

[15] P.L. Erdős - K. Rice - M.A. Steel - L.A. Székely - T.J. Warnow: The Short Quartet Method, to appear in Math. Modelling and Sci. Computing Special Issue of the papers presented at the Computational Biology sessions at the 11th ICMCM, March 31 - April 2, 1997, Georgetown University Conference Center, Washington, D.C., USA.

[16] P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: Constructing big trees from short sequences, *Automata, Languages and Programming* 24th International Colloquium, ICALP'97, Bologna, Italy, July 7 - 11, 1997, (P. Degano,; R. Gorrieri, A. Marchetti-Spaccamela, Eds.) Proceedings (Lecture Notes in Computer Science. Vol. 1256) (1997), 827-837.

[17] **P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: A few logs suffice to build (almost) all trees (I),** *Random Structures and Algorithms* **14 (1999), 153-184.**

[18] **P.L. Erdős - M.A. Steel - L.A. Székely - T.J. Warnow: A few logs suffice to build (almost) all trees (II),** *Theoretical Computer Science*, **221 (1-2) (1999), 77–118.**

[19] P.L. Erdős - P. Sziklai - D. C. Torney: A finite word poset, *Electr. J. Combinatorics*, **8** No 2. (2001), R# 8.

[20] A.W.M. Dress - P.L. Erdős: X-trees and Weighted Quartet Systems, *Ann. Combin.* **7 (2003), 155-169**

[21] A.G. D'yachkov - P.L. Erdős - A.J. Macula - V.V. Rykov - D.C. Torney - C-S. Tung - P.A. Vilenkin - P. Scott White: Exordium for DNA Codes, *J. Comb. Opt.* **7** (4) (2003), 369–379.

[22] A.W.M. Dress - P.L. Erdős: Reconstructing Words from Subwords in Linear Time, *Annals of Combinatorics*, **8** (4) (2004), 457–462.

[23] P.L. Erdős - P. Ligeti - P. Sziklai - D.C. Torney: Subwords in reverse complement order - extended abstract, invited paper to *Proc. Conf. on "Combinatorial and Algorithmic Foundations of Pattern and Association Discovery" - Schloss Dagstuhl, International Conference And Research Center For Computer Science, Germany* May 14-19. 2006, 1–7.

[24] A. Apostolico - P.L. Erdős - M. Lewenstein: Parameterized Matching with Mismatches, *J. of Discrete Algorithms* **5** (2007), 135–140.

[25] P.L. Erdős - P. Ligeti - P. Sziklai - D.C. Torney: Subwords in reverse complement order, *Annals of Combinatorics* **10 (2006) 415–430.**

Hivatkozott idegen cikkek

[AhlKha00] R. Ahlswede - L. Khachatrian: Splitting properties in partially ordered sets and set systems, in *Numbers, Information and Complexity* (Althöfer et. al. editors) Kluver Academic Publisher, (2000), 29-44.

[AllRho04] E.S. Allman - J.A. Rhodes: Quartets and Parameter Recovery for the General Markov Model of Sequence Mutation, *AMRX App. Math. Res. Express* (2004), 107–131.

[AllRho06] E.S. Allman - J.A. Rhodes: The identifiability of tree topology for phylogenetic models, including covarion and mixture models, *J. Comp. Biol.* **13** (5) (2006), 1101–1113.

[Att99] K. Atteson: The performance of neighbor-joining methods of phylogenetic reconstruction, *Algorithmica* **25** (1999), 251–278.

[Ber08] F. Bernstein: Zur Theorie der triginomischen Reihen, *Leipz. Ber* (Berichte über die Verhandlungen der Königl. Sächsischen Gesellschaft der Wissenschaften zu Leipzig. Math.-phys. Klasse) **60** (1908), 325–338

[BerKer99] V. Berry - Tao Jiang - P. Kearney - Mi Li - T. Wareham: Quartet cleaning: improved algorithms and simulations, *Algorithms – ESA'99, 7th European Symposium on Algorithms* Prague, Chezh Rep. Lect. Notes Comp. Sci **1643** (1999), 313–324.

[Bry05] D. Bryant: Extending tree models to split networks, *Chapter 17, in Algebraic Statistics for Computational Biology* (Ed. L. Pachter and B. Sturmfels) Cambridge Univ. Press (2005), 331–346.

[Bun71] P. Buneman: The recovery of trees from measures of dissimilarity, in *Mathematics in the Archaeological and Historical Sciences*, F. R. Hodson, D. G. Kendall, P. Tautu, eds.; Edinburgh University Press, Edinburgh, 1971, 387–395.

[BurFra90] G. Burosch, U. Franke, S. Röhl: Über Ordnungen von Binärworten, *Rostock. Math. Kolloq.* **39** (1990), 53–64.

[BurGro96] G. Burosch, H-D. Gronau, J-M. Laborde: On posets of m-ary words, *Discrete Math.* **152** (1996), 69–91.

[CarHen90] M. Carter - M. Hendy - D. Penny - L. A. Székely - N.C. Wormald: On the distribution of lengths of evolutionary trees, *SIAM J. Disc. Math.* **3** (1990), 38-47.

[Cha76] P.J. Chase: Subsequence numbers and logarithmic concavity, *Discrete Math.* **16** (1976), 123–140.

[ChoTul05] B. Chor - T. Tuller: Maximum likelihood of evolutionary trees: hardness and approximation, *Bioinformatics* **21** Suppl.1 (2005), I97–I106.

[CowKol06] R. Cowen - A. Kolany: Davis-Putman style rules for deciding Property S, submitted (2006), 1–10.

[CsuKao99] M. Csűrös - M-Y. Kao: Recovering evolutionary trees through Harmonic Greedy Triplets. *SODA '99 - Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, (1999), 261–270.

[DahJoh92] E. Dahlhaus - D.S. Johnson - C.H. Papadimitriou - P.D. Seymour - M. Yannakakis: The complexity of multiway cuts, 24^{th} *ACM STOC*, (Editors: Rao Kosaraju , Mike Fellows , Avi Wigderson , John Ellis) (1992), 241–251.

[DahJon94] E. Dahlhaus - D.S. Johnson - C.H. Papadimitriou - P.D. Seymour - M. Yannakakis: The complexity of multiterminal cuts, *SIAM J. Computing* **23** (1994), 864–894.

[DasHil06] C. Daskalakis - C. Hill - A. Jaffe - R.H. Mihaescu - E. Mossel - S. Rao: Maximal accurate forests from distance matrices, *RECOMB'06* **LNCS 3909** (2006), 281–295.

[DasMos06] C. Daskalakis - E. Mossel - S. Roch: Optimal phylogenetic reconstruction, *Proceedings of ACM STOC'06* (2006), 159–168.

[DriAne04] A.C. Driskell - C. Ané - J.G. Burleigh - M.M. McMahon - B.C. O'Meara - M. J. Sanderson: Prospects for Building the Tree of Life from Large Sequence Databases, *SCIENCE* **306** (5699) (2004), 1172–1174.

[DufSan01] D, Duffus - W. Sands: Minimum sized fibres in distributive lattices, *Austr. J. Math* **70** (2001), 337–350.

[DufSan03] D, Duffus - W. Sands: Finite distributive lattices and the splitting property, *Algebra Universalis* **49** (2003), 13–33.

[DufSan05] D. Duffus - B. Sands: Splitting numbers of grids, *Elec. J. Comb.* **12** (2005), R#17

[DyaMac05] A.G. D'yachkov - A.J. Macula - W.K. Pogozelski - T.E. Renz - V.V. Rykov - D.C. Torney: A weighted insertion-deletion stacked pair thermodynamic metric for DNA codes, *DNA Computing LNCS* **3384** (2005), 90-103.

[DyaVil05] A. G. D'yachkov - P.A. Vilenkin - I. K. Ismagilov - R. S. Sarbaev - A. Macula - D. Torney - S. White: On DNA Codes, *Problems of Information Transmission* **41** (2005), 349–367. (Originally published in *Problemy Peredachi Informatsii*, No. 4, (2005), 57–77.)

[Dza92] Mirna Džamonja: Note on splitting property in strongly dense posets of size $\aleph_0$, *Radovi Matematički* **8** (1992), 321-326.

[EmlMar05] D.J. Emlen - J. Marangelo - B. Ball - C.W. Cunningham: Diversity in the weapons of sexual selection: Horn evolution in the beetle genus Onthophagus (Coleoptera: Scarabaeidae). *Evolution* **59** (2005), 1060–1084.

[EriRan04] N. Eriksson - K. Ranestad - B. Sturmfels - S. Sullivant: Phylogenetic algebraic geometry, in *in "Projective Varieties with Unexpected Properties" A Volume in Memory of Giuseppe Veronese. Proceedings of the international conference "Varieties with Unexpected Properties"*, Siena, Italy, June 8-13, 2004 (Ed. by Ciliberto, Ciro; Geramita, Antony V.; et al.) (2005), 237–258.

[EvaSpe93] S.N. Evans - T.P. Speed, Invariants of some probability models used in phylogenetic inference, *Annals of Statistics*, **21** (1993), 355–377.

[Fel03] J. Felsenstein: *Inferring Phylogenies*, Sinauer Associates, Ins. Sunderland, Massachusetts, 2003. pp. 664.

[Gus97] D. Gusfield: *Algorithms on strings, trees and sequences*, Cambridge University Press, 1997.

[GraFou82] R.L. Graham and L.R. Foulds: Unlikelihood that minimal phylogenies for a realistic biological study can be constructed in reasonable computational time, *Math. Biosci.* **60** (1982), 133–142.

[HasMan98] W. Hasan - R. Motwani: Coloring away communication in parallel query optimization, *Proc. 21st VLDB Conf. Zürich, Switzerland, (1995)* Readings in Database Systems, 3rd Edition (Michael Stonebraker, Joseph M. Hellerstein, eds.) Morgan-Kaufmann Publishers, (1998) 239–250.

[HelNes04] P. Hell - J. Nešetril: *Graphs and homomorphisms*, Oxford Lecture Series in Math. and Appl. **28**, (2004), pp. 244.

[HenPen93] M.A. Hendy - D. Penny: Spectral analysis of phylogenetic data, *J. Classification.* **10** (1993), 1–10.

[HofKom76] G. Hoffmann - P. Komjáth: The transversal property implies property B, *Periodica Math. Hung.* **7** (1976), 179–181.

[HusNet98] D. Huson - S. Nettles - L. Parida - T. Warnow - S. Yooseph, The Disk-Covering Method for Tree Reconstruction, Proceedings of *Proc. "Algorithms and Experiments"*, (ALEX'98), Trento, Italy (1998), 62–75.

[JarBas01] P.D. Jarvis - Bashford J.P.: Quantum field theory and phylogenetic branching, *J. Physics A - Mathematical and General* **34 (49)** (2001), L703–707.

[Lak87] J.A: Lake: A rate-independent technique for analysis of nucleic acid sequences: Evolutionary parsimony, *Mol. Bio. Evol* **4** (1987), 167–191.

[LanRob04] B. Landman - A. Robertson: *Ramsey theory on the Integers*, AMS Student Math. Library Vol. 24 (2004), Chapter 2.

[Lev92] V. Levenshtein: On perfect codes in deletion and insertion metric, *Discrete Math. Appl.* **2** (1992), 241–258.

[Lev01a] V.I. Levenshtein: Efficient reconstruction of sequences from their subsequences or supersequences, *J. Comb. Theory* (A) **93** (2001), 310–332.

[Lev01b] V.I. Levenshtein: Efficient reconstruction of sequences, *IEEE Tr. Inf. Theory* **47** (1) (2001), 2–22.

[LigSzi05] P. Ligeti - P. Sziklai: Automorphism of subword-posets, *Disc. Math.* **503** (2005), 372–378.

[Lot97] M. Lothaire : *Combinatorics on words,* Cambridge University Press, Cambridge, 1997.

[Lov79] Lovász László: *Combinatorial Problems and Exercises,* North Holland, 1979.

[Mac03] A.J. Macula: DNA Tag-Antitags (TAT) codes, *US Air Force AFRL-IF-RS-TR-2003-57* (2003), 1–23.

[MilKas05] O. Milenkovic - N. Kashyap - B.Vasic: On DNA Computers Controlling Gene Expression Levels, invited talk in *44th IEEE Conf.on Decision and Control CDC-ECC'05* (2005), 1770–1775.

[Mil37] E.W. Miller: On a property of families of sets, *C. R. Soc. Sci. Varsovie* **30** (1937), 31-38

[Mor96] D.A. Morrison: Phylogenetic tree-building, *Int. J. Parasitology* **26** (1996), 589–617.

[Mos03] E. Mossel: On the impossibility of reconstructing ancestral data and phylogenies, *J. Comp. Biol.* **10** (2003), 669–676.

[Mos04] E. Mossel: Phase transitions in phylogeny , *Transactions of the AMS* **356** (2004), 2379–2404.

[MosRoc05] E. Mossel - S. Roch: Learning nonsingular phylogenies and hidden Markov models, *Proceedings of ACM STOC'05* (2005), 366–375.

[MosVig05] E. Mossel - E. Vigoda: Phylogenetic MCMC algorithms are misleading on mixtures of trees, *Science* **309** (2005), 2207–2209. Online supporting material

[MosVig06] E. Mossel - E. Vigoda: Response to Comment on "Phylogenetic MCMC algorithms misleading on mixture of trees, *Science* **312** (2006), 367b.

[NguSpe92] T. Nguyen - T.P. Speed: A derivation of all linear invariants for a non-balanced transversion model, *J. Mol. Evol* **35** (1992), 60–76.

[NolMan06] J.P. Nolan - F. Mandy: Multiplexed and microparticle-based analysis: Quantitative tools for the large-scale analysis of biological systems, *CYTOMETRY PART A* **69A** (2006), 318–325.

[PatWal00] A.M. Paterson - L.J. Wallis - G.P. Wallis: Preliminary molecular analysis of Pelecanoides georgicus (Procellariiformes: Pelecanoididae) on Wheuna Hou (Codfish Island): implication for its taxonomic status, *New Zealand J. Zoology* **27** (2000), 415–423.

[PenLoc94] D. Penny - P.J. Lockhart - M.A. Steel - M.D. Hendy: The role of models in reconstructing evolutionary trees, in *Models in Phylogeny Reconstructions* (ed. R.W. Scotland, D.J. Siebert and D.M. Williams), Systematics Association Special Volume **52** Clarendon Press, Oxford (1994), 211–230.

[Pou06] M. Pouly: Minimizing Communication Costs of Distributed Local Computation., in *ECAI'2006, Workshop 26: Inference methods based on graphical structures of knowledge* (ed. A. Darwiche and R. Dechter and H. Fargier and J. Kohlas and J. Mengin and G. Verfaillie and N. Wilson), (2006), 19–24.

[Rob03] F.S. Roberts: Challenges for Discrete Mathematics and Theoretical Computer Science in the Defense against Bioterrorism, in *Bioterrorism: Mathematical Modeling Applications in Homeland Security* (ed. by H. T. Banks and Carlos Castillo-Chavez), Proceeding of DIMACS and NSF, 2002, SIAM (2003), Chapter 1.

[RokCar05] A. Rokas - S.B. Caroll: More gens or more taxa? The relative contribution of gene number and taxon number to phylogenetic accuracy, *Mol. Biol. Evol.* **22** (2005), 1337–1344.

[San93] D. Sankoff, Analytical approaches to genomic evolution, *Biochemie* **75** (1993) (5), 409–413.

[SemSte03] C. Semple - M.A. Steel: *Phylogenetics*, Oxford Lecture Series in Mathematics and Its Applications 24. Oxford University Press 2003. pp. 239.

[Sim75] I. Simon: Piecewise testable events, (H. Brakhage ed.), Automata Theory and Formal Languages, LNCS. **33**, Springer Verlag, (1975), 214–222.

[Steel93] M.A. Steel: Decomposition of leaf-colored binary trees, *Advances in Appl. Math* **14** (1993), 1–24.

[StrHae96] K. Strimmer - A. von Haeseler: Quartet Puzzling: a quartet Maximum Likelihood method for reconstructing tree topologies, *Mol. Biol. Evol.*, **13** (1996), 964–969.

[SwoOls96] D.L. Swofford - G.J. Olsen - P.J. Waddell - D.M. Hillis, Phylogenetic Inference, in *Molecular Systematic, Second Edition* D.M. Hillis, C. Moritz, B.K. Mable (eds.), Sinauer Associates, Inc. Publishers, Sunderland, Massachusetts, USA 1996.

[Wil04] S.J. Willson: Constructing rooted supertrees using distances *Bulletin of Mathematical Biology* **66** (2004), 1755–1783.

[WuLin04] Gang Wu - Guohui Lin - Jia-Huai You: Quartet Based Phylogeny Reconstruction with Answer Set Programming, in *16th IEEE Int. Conf. on Tools with Artificial Intelligence (ICTAI'04)* (2004), 612–619.

A szerző egyéb cikkei

[26] Erdős Péter: Egy Ramsey-típusú tétel, *Matematikai Lapok,* **27** (1976–79), 361–364.

[27] P.L. Erdős - Z. Füredi: On automorphisms of line-graphs, *Europ. J. Combinatorics* **1** (1980), 341-345.

[28] P.L. Erdős - P. Frankl - G.O.H. Katona: Intersecting Sperner families and their convex hulls, *Combinatorica* **4** (1984), 21-34.

[29] P.L. Erdős - P. Frankl - G.O.H. Katona: Extremal hypergraphs problems and convex hulls, *Combinatorica* **5** (1985), 11-26.

[30] P.L. Erdős - E. Győri: Any four independent edges of a 4-connected graph are contained in a circuit. *Acta Math. Sci. Hung.* **46** (1985), 311-313.

[31] P.L. Erdős - G.O.H. Katona: Convex hulls of more-part Sperner families, *Graphs and Combinatorics* **2** (1986), 123-134.

[32] P.L. Erdős - G.O.H. Katona: All maximum 2-part Sperner families, *J. Combinatorial Theory* (A) **43** (1986), 58-69.

[33] P.L. Erdős - G.O.H. Katona: A 3-part Sperner theorem, *Studia Scientiarum Mathematicarum Hungarica* **22** (1987), 383-393.

[34] P.L. Erdős - K. Engel: Sperner families satisfying additional conditions and their convex hulls, *Graphs and Combinatorics* **5** (1988), 50-59.

[35] P.L. Erdős - L.A. Székely: Applications of antilexicographical order I. An enumerative theory of trees, *Advances in Applied Mathematics* **10** (1989), 488-496.

[36] K. Engel - P.L. Erdős: Polytopes determined by complementfree Sperner families, *Discrete Mathematics* **81** (1990), 165-169.

[37] P.L. Erdős - P. Frankl - D.J. Kleitman - M. Saks - L.A. Székely: Sharpening the LYM inequality, *Combinatorica* **12** (1992) 295-301.

[38] P.L. Erdős - U. Faigle - W. Kern: A group-theoretic setting for some intersecting Sperner families, *Combinatorics, Probability and Computing* **1** (1992), 323-334.

[39] P.L. Erdős - Niall Graham: On maximal Sperner families, *DIMACS Technical Report*, **TR 93-42** Rutgers University, New Jersey, USA

[40] P.L. Erdős - L.A. Székely - Á. Seress: On intersecting chains in Boolean algebras, *Combinatorics, Probability and Computing* **3** (1994), 57–62.

[41] P.L. Erdős: On the reconstruction of combinatorial structures from line-graphs, *Studia Scientiarum Math. Hung* **29** (1994), 341-347.

[42] R. Ahlswede - P.L. Erdős - Niall Graham: A splitting property of maximal antichains, *Combinatorica* **15** (1995), 475-480.

[43] P.L. Erdős - U. Faigle - W. Kern: On the average rank of LYM-sets, *Discrete Mathematics* **144** (1995), 11-22.

[44] P.L. Erdős: Splitting property in infinite posets, *Discrete Mathematics* **163** (1997), 251–256.

[45] R. Ahlswede - N. Alon - P.L. Erdős - M. Ruszinko - L.A. Székely: Intersecting systems, *Combinatorics, Probability and Computing* **6**(2)(1997), 127–137.

[46] P.L. Erdős - L.A. Székely: Pseudo-LYM inequality and AZ identities, *Adv. Appl. Math* **19** (1997), 431-443.

[47] P.L. Erdős - L.A. Székely - Á. Seress: On intersecting chains in Boolean algebras, in *Combinatorics, geometry and probability* (ed. B. Bollobás, A. Thomason) (Cambridge, 1993), Cambridge Univ. Press, Cambridge, 1997. 299–304. Second release

[48] P.L. Erdős: Some generalizations of property B and the splitting property, *Annals of Combinatorics* **3** (1999), 53–59.

[49] P.L. Erdős - Á. Seress - L.A. Székely: Erdős-Ko-Rado and Hilton-Milner type theorems for intersecting chains in posets, *Combinatorica* **20** (2000), 27–45.

[50] P.L. Erdős - L.A. Székely: Erdős-Ko-Rado theorems of higher order, in *Numbers, Information and Complexity*, (I. Alth"ofer, Ning Cai, G. Dueck, L. Khachatrian, M. S. Pinsker, A. Sark"ozy, I. Wegener and Zhen Zhang (eds.)), Kluwer Academic Publishers (2000), 117–124.

[51] P.L. Erdős - U. Faigle - W. Hochstätter - W. Kern: Note on the Game Chromatic Index of Trees, *Theoretical Computer Science*, (Special Issue on Algorithmic Combinatorial Game Theory) **313** (3) (2004), 371–376.

[52] P.L. Erdős - Z. Füredi - G.O.H. Katona: Two part and k-Sperner families - new proofs using permutations, *SIAM J. Discrete Math.* **19** (2005), 489–500.

[53] P.L. Erdős - Á. Seress - L.A. Székely: Non-trivial t-intersection in the function lattice, *Annals of Comb.* **9** (2005), 177–187.

[54] H. Aydinian - P.L. Erdős: All maximum size 2-part Sperner systems - in short, *Comb. Prob. Comp.* **16** (4) (2007), 553–555.

[55] P.L. Erdős - L. Soukup: How to split antichains in infinite posets, *Combinatorica* **27** (2) (2007), 147–161.

[56] D. Duffus - P.L. Erdős - J. Neŝetril - L. Soukup: Splitting property in the graph homomorphism poset, to appear in *Comment Math Univ Carolinae* (2007), 1–12.

Előkészületben

[57] P.L. Erdős - L. Soukup: Quasikernels in infinite graphs, submitted (2007), 1–17.

[58] A. Apostolico - P.L. Erdős - A. Jüttner - A. Sali: Parameterized Matching with Mismatches in case of general alphabets, in preparation (2006).

[59] H. Aydinian - P.L. Erdős - L.A. Székely: 2-part L-Sperner families, in preparation (2006), 1–17.

Discrete Applied Mathematics 47 (1993) 1–8
North-Holland

Counting bichromatic evolutionary trees

Péter L. Erdős*

Hungarian Academy of Sciences, Budapest, Hungary; and Institute für Ökonometrie und Operations Research, Rheinische Friedrich-Wilhelms Universität, Bonn, Germany

L.A. Székely*

Department of Computer Science, Eötvös L. University, Budapest, Hungary; and Institute für Ökonometrie und Operations Research, Rheinische Friedrich-Wilhelms Universität, Bonn, Germany

Received 13 December 1990
Revised 17 September 1993

Abstract

We give a short and transparent bijective proof of the bichromatic binary tree theorem of Carter, Hendy, Penny, Székely and Wormald on the number of bichromatic evolutionary trees. The proof simplifies M.A. Steel's proof.

Evolutionary trees are extensively studied structures in biostatistics. (These are leaf-coloured binary trees. For details see, e.g., Felsenstein [4], Steel [10] or Carter et al. [1].)

In general, the mathematical problems arising here are hard (see [6]). One of the very beginning steps is to count evolutionary trees. For two colours it was done by Carter et al. [1]. Their work is based on the generating function method and on a lengthy, computer-assisted application of the multivariate Lagrange inversion. Recently Steel [10] gave a bijective proof for the bichromatic binary tree theorem pioneering the application of Menger's theorem in enumerative theory. Unfortunately, his solution is rather involved. The goal of the present paper is to give a simple and transparent bijective proof for the bichromatic binary tree theorem. Our work was inspired by Steel's work, actually we simplify some crucial steps in his proof and the rest of the proof is identical to his one. The proof uses more graph theory than proofs in enumerative theory usually do.

Correspondence to: Professor P.L. Erdős, Hortensiastraat 3, 1338 ZP Almere, Netherlands.
* Research supported in part by Alexander v. Humboldt-Stiftung.

Preliminaries and the bichromatic binary tree theorem

In this section we introduce some definitions and notations which may not be common, and state the theorem of Carter et al.

In a tree, a vertex of degree 1 is a *leaf*. A tree is *binary* if every nonleaf vertex of the tree has degree 3. A tree is *rooteed binary* if it has exactly one vertex of degree 2 and the other nonleaf vertices have degree 3. The vertex of degree 2 is the *root* of the tree. By definition, a singleton vertex is a binary tree and also a rooted binary tree. In this degenerate tree above, the singleton vertex is a leaf, and in the rooted case it is a root as well.

A (rooted) binary tree with labelled leaves is termed a (rooted) *semilabelled* tree. Hereafter we identify the set of leaves and the set of labels and denote both by L. A *semilabelled rooted binary forest* is a forest containing rooted semilabelled binary trees, where the label sets of distinct trees are pairwise disjoint. The following facts are well known. (The details can be found in several books and papers, e.g., see [1, 2, 3].)

Lemma 0. (a) *Any binary tree T with n leaves has $2n-2$ vertices and $2n-3$ edges.*

(b) *Any rooted binary tree T with n leaves has $N(T) = 2n-1$ vertices and $2n-2$ edges.*

(c) *The total number of semilabelled binary trees with n leaves is*

$$b(n) = (2n-5)!!.$$

(d) *The total number of semilabelled rooted binary forests with n leaves and k trees is*

$$N(n,k) = \binom{2n-k-1}{k-1}(2n-2k-1)!!.$$

Let T be a semilabelled binary tree. We term a map $\chi: L \to \{A, B\}$ a *leaf-colouration*. A colouration $\bar{\chi}: V(T) \to \{A, B\}$ is an *extension* of the leaf-colouration χ if the two maps are identical on the set L. The *changing number* of the colouration $\bar{\chi}$ is the number of edges whose endvertices have different colours according to $\bar{\chi}$. An extension is a *minimal colouration* according to the leaf-colouration χ if its changing number is minimal among the changing numbers of all extensions of χ. We refer to the minimal changing number as the *length* of the tree T (according to χ). An efficient algorithm for calculating the length of a tree and finding a minimal colouration, due to [5], is established in [7].

Let us fix now a 2-colouration χ of the set L and denote by L_A and L_B the nonempty colour classes ($L_A \cup L_B = L$). Set $a = |L_A| > 0$ and $b = |L_B| > 0$. The question is: What is the number of (unrooted) semilabelled binary trees whose leaf set is L and length is exactly k (according to χ)? Let $f_k(a, b)$ denote the number in question. Carter, Hendy, Penny, Székely and Wormald proved [1], that

Theorem.

$$f_k(a,b) = (k-1)!\,(2n-3k)\,N(a,k)\,N(b,k)\,\frac{b(n)}{b(n-k+2)}$$

where $a + b = n$, $a > 0$, $b > 0$.

In the rest of our paper we prove this theorem. The proof is based on a method developed by Steel [10].

Steel's decomposition

In this section we describe the structure of the bichromatic semilabelled trees of length k.

Let χ be a 2-colouration of the set L. The length of the tree T is equal to k iff the deletion of k well-chosen edges decomposes T into subtrees with one colour being present in each, but the deletion of less than k edges cannot do it. Due to Menger's theorem [8], this means that the maximum number of edge-disjoint paths from L_A to L_B is k. Since T is binary, two edge-disjoint paths between leaves are also vertex-disjoint. Therefore there exist k (but no more than k) vertex-disjoint paths from L_A to L_B. A second application of Menger's theorem guarantees the existence of a k-element vertex set which covers *every* $L_A \to L_B$ path. Any such set is called a *minimal covering system.* It is easy to see that incidence defines a one-to-one correspondence between any minimal covering system and any k vertex-disjoint paths from L_A to L_B.

The following lemma helps to understand the minimal covering systems.

Lemma 1. *Suppose* M *is a minimal covering system. Set*

$$\mu(T) = \left\{ \bigcap_{\pi \in \Pi} \{P\colon m \in P \in \pi\}\colon m \in M \right\},$$

where Π *is the family of sets of* k *edge-disjoint paths connecting* L_A *and* L_B. *Then*

(a) $\mu(T)$ *is independent of the choice of* M, *the members of* $\mu(T)$ *are vertex-disjoint paths in* T.

(b) *Assume* $v_0 \in \bigcup \mu(T)$. *Define the set* M_0 *by picking the vertex closest to* v_0 *from every path of* $\mu(T)$. *Then* M_0 *is a minimal covering system, hence, any point of any member of* $\mu(T)$ *belongs to some minimal covering system.*

(c) $v_0 \in M_0$ *and* M_0 *is unique as long as* v_0 *is given.*

Proof. Notice the following consequence of Menger's theorem: for minimal covering systems M', M'', a set of k edge-disjoint paths from L_A to L_B defines a matching between M' and M'' by the relation "being on the same path".

To prove (a), we have to see that any set of k edge-disjoint paths from L_A to L_B define the *same* matching.

On the contrary, assume that two path systems define two different matchings of M', M''. The two matchings define a graph G on the vertex set $M' \triangle M''$ with edges taken from the matchings. G contains a cycle of length longer than 2. Recall that the edges of this cycle can be represented by subpaths of the two path systems. Since T is cycle-free, these subpaths altogether cover twice a path P of T. This contradicts to the disjointness of the path systems.

We have proved that $\mu(T)$ is independent of the choice of M. Finally, note that a nonempty intersection of paths in a tree is a path itself.

(We do not need this explicitly, but you may observe that any system of representatives of $\mu(T)$ covers every path of every π and clearly every minimal covering system M occurs as such a system of representatives—just define $\mu(T)$ by this M! Unfortunately, not every system of representatives is a minimal covering system. This makes life more difficult.)

To prove (b) notice that every $L_A \to L_B$ path intersects at least one member of $\mu(T)$. If a path P' from L_A to L_B intersects two members of $\mu(T)$, then one member separates the other member from v_0. Now by definition, the first intersection of P' with the other member belongs to M_0 and covers the path P'. Hence we may assume that P' intersects a unique $P \in \mu(T)$. We claim that P' contains the whole P. Hence $P \cap M_0 \in P'$.

In order to prove the latter claim, we consider two cases. Either $P' \in \pi$ for some $\pi \in \Pi$, or not. In the first case, P' occurs in the intersection that defines P, hence $P \subset P'$. In the second case, P' intersects two paths from every $\pi \in \Pi$, otherwise we may exchange P' with the only path π intersected by P' to get a $P' \in \pi' \in \Pi$. It is easy to conclude that there exist $P_1, P_2 \in \mu(T)$, such that P' intersects two paths from every π, which contain P_1, P_2, respectively. Finally, P' intersects both P_1, P_2, a contradiction. □

Take M_0 from Lemma 1. Define the semilabelled forest $\mathcal{F}' = \{T'_v: v \in M_0\}$ of pairwise disjoint subtrees of T as follows: For every vertex u of the tree T the unique path $u \to v_0$ contains at least one element of M_0. Let u belong to T'_v iff v is the nearest vertex to u among these vertices. Finally, let the tree T_v ($v \in M_0$) be the subtree of T'_v which is spanned by those leaves of T'_v which also belong to L.

Lemma 2. *The semilabelled forest $\mathcal{F} = \{T_v: v \in M_0\}$ satisfies the following conditions:*

(a) *The leaf set of $\mathcal{F}$ coincides with L.*

(b) *If $v \in M_0$ then $v \in T_v$ and the path $v_0 \to T_v$ reaches the tree T_v at the vertex v.*

(c) *The degree of the vertex $v \in (M_0 \setminus \{v_0\})$ in the tree T_v is equal to 2.*

(d) *Every tree T_v is bichromatic (that is it has two colours) according to the leaf-colouration χ. Removing the vertex v from the tree T_v, the remaining two (or if $v = v_0$, then two or three) subtrees are monochromatic according to χ.*

Proof. Parts (a) and (b) directly follow from the definition of $\mathcal{F}$. Part (c) follows from (b). Part (d) contains the essence of this lemma. The set M_0 is a covering system, therefore the subtrees derived by removing the vertex v must be monochromatic (i.e., they cannot contain leaves of different colours). On the other hand, these subtrees must show two different colours, otherwise any path $P: L_A \to L_B$ covered solely by vertex v out of the elements of M_0 must be closer to the vertex v_0 than the subtree T_v itself. Therefore the neighbour v' of vertex v in the direction of v_0 also covers P. So the choice of v from M_0 was wrong, v' must have been chosen. □

In the next step we derive a new semilabelled forest from $\mathcal{F}$: for every vertex $v \in M_0$ we contract the vertices of degree 2 in the tree T_v, except the vertex v itself. Finally if the degree of v_0 in the tree T_{v_0} is equal to 3 then we add a root into this tree which covers every $L_A \to L_B$ path in T_{v_0}. Denote $\mathcal{F}^S$ the derived semilabelled forest consisting of k rooted binary trees. This forest is the *Steel decomposition* of the tree T (with respect to the leaf-colouration χ and the vertex v_0). We call the tree derived from T_{v_0} the *kernel* of that decomposition.

Lemma 3. *For any given v_0, the Steel decomposition of the tree T is unique. Moreover, if $v_0, v_0' \in P \in \mu(T)$, then they define the same Steel decomposition.*

Proof. By definition, the forest $\mathcal{F}^S$ is determined by the minimal covering system M_0. We have already proved the uniqueness of M_0. Changing v_0 for v_0', we end up with $M_0' = M_0 - \{v_0\} \cup \{v_0'\}$. □

Let $\mathcal{F} = \{T_0; T_1, \ldots, T_{k-1}\}$ be an arbitrary semilabelled rooted binary forest with leaf set $L = L_A \cup L_B$. Let e_i $(i = 1, \ldots, k-1)$ denote the number of edges in the tree T_i, and let e_0 be (edge number of T_0) $- 1$. An *extension* of the forest $\mathcal{F}$ is a semilabelled binary tree whose Steel decomposition is the forest $\mathcal{F}$ with kernel T_0.

The first question is: How can we find extensions of the forest $\mathcal{F}$? Let B be a binary tree and let B_1 be a rooted binary tree. The *insertion* of B_1 into B is the following operation: subdivide by a new vertex one of the edges of B and connect the new vertex to the root of B_1 by a new edge.

Lemma 4. *Let $\mathcal{F} = \{T_0; T_1, \ldots, T_{k-1}\}$ be a semilabelled rooted binary forest. Let $\hat{T}_0$ be the binary tree derived from T_0 by deleting the root and joining its neighbours. Insert recursively the trees $T_1, T_2, \ldots, T_{k-1}$ into the actual tree, where the initial actual tree is $\hat{T}_0$, and later on the actual tree is the result of the last insertion. Let T be the semilabelled binary tree which is the last actual tree. Then there is a vertex v_0 in T, such that the Steel decomposition of the tree T according to v_0 coincides with the forest $\mathcal{F}$.*

Proof. Let v_0 be any neighbour of the root of T_0 in $\hat{T}_0$. This vertex covers every path $L_A \to L_B$ in the tree $\hat{T}_0$. The vertex v_0 together with the original roots of $T_1, \ldots, T_{k-1}$ form a minimal covering system in the tree T. It is easy to see that this system also

satisfies the minimum distance condition with respect to the vertex v_0. Therefore the Steel decomposition of T with respect to v_0 is $\mathscr{F}$. □

Lemma 5. *Let* $\mathrm{Ext}(T_0; T_1, \dots, T_{k-1})$ *denote the set of extensions of the forest* $\mathscr{F}$. *We have*

$$|\mathrm{Ext}(T_0; T_1, \dots, T_{k-1})| = e_0 \frac{b(n)}{b(n-k+2)}.$$

Proof. We apply mathematical induction on k. If we use the abbreviation $T(e_0, k-1) = |\mathrm{Ext}(T_0; T_1, \dots, T_{k-1})|$, then we have to prove, that:

(a) $T(e_0, 1) = 1$;

(b) $T(e_0, k-1) = (2n-2k+1)\,T(e_0, k-2)$.

Case (a) is trivial, because the unique extension of the forest $\{T_0\}$ is the tree $\hat{T}_0$ itself.

(b) Suppose T is an extension of $\mathscr{F}$. Define a directed tree T^c as follows: The vertices of T^c are $\hat{T}_0, T_1, \dots, T_{k-1}$. An arbitrary ordered pair (T_i, T_j) (or $(\hat{T}_0, T_j)$) is an arc if the last root of the trees $\hat{T}_0, T_1, \dots, T_{k-1}$ before v_j on the path $v_0 \to v_j$ in the tree T is the vertex v_i. Every vertex of T^c (except the vertex $\hat{T}_0$) has in degree exactly one, and the corresponding arc tells us where the tree T_j is inserted in this extension. Examine the insertion of the tree T_1. We distinguish two disjoint subcases:

(b1) There is an $i \in \{2, \dots, k-1\}$ for which (T_i, T_1) is an arc in T^c. Then there are e_i different insertions of T_1 into T_i. After any of these insertions we have a forest of $k-1$ trees (one of them is the kernel T_0). By the inductive hypothesis any forest built has $T(e_0, k-2)$ different extensions. So the total number of extensions of these types is

$$(e_2 + e_3 + \cdots + e_{k-1})\,T(e_0, k-2).$$

(b2) The ordered pair (T_0, T_1) is an arc in T^c. In this case the tree T_1 is inserted into the tree $\hat{T}_0$. We have e_0 different ways to realize this insertion. After the insertion we have a forest of $k-1$ trees, where the kernel has $e_0 + e_1 + 2$ edges. Therefore any of the forests built can be extended in

$$(e_0 + e_1 + 2)\frac{b(n)}{b(n-[k-1]+2)}$$

ways. Therefore the total number of extensions of this type is

$$(e_0 + e_1 + 2)\,T(e_0, k-2).$$

Adding up the numbers from the subcases, the total number of the extensions is

$$\begin{aligned} T(e_0, k-1) &= (e_0 + e_1 + \cdots + e_{k-1} + 2)\,T(e_0, k-2) \\ &= (2n-2k+1)\,T(e_0, k-2). \end{aligned}$$

(In the last step we used Lemma 0(a) and (b).) □

The proof of the Theorem

Let χ be an arbitrary but fixed 2-colouration of the set L with colour classes L_A and L_B, where $|L_A| = a$ and $|L_B| = b$. Denote $\mathscr{F}_k(a,b)$ the set of semilabelled binary trees of length k (according to χ) with leaf set L. Let

$$\mathscr{F}_k^*(a,b) = \{(T,P)\colon T \in \mathscr{F}_k(a,b),\ P \in \mu(T)\}.$$

Let $\mathscr{H}(a,b,k)$ denote the collection of semilabelled rooted binary forests of k trees with leaf set L, such that every tree has two oppositely coloured, monochromatic subtrees if its root is removed. Finally let

$$\mathscr{G}_k(a,b) = \{(\mathscr{F}, T_0, T)\colon \mathscr{F} \in \mathscr{H}(a,b,k),\ T_0 \in \mathscr{F}, T \in \mathrm{Ext}(T_0; \mathscr{F} \setminus \{T_0\})\}.$$

Lemma 6. *There exists a bijection ψ from $\mathscr{F}_k^*(a,b)$ onto $\mathscr{G}_k(a,b)$.*

Proof. For $(T,P) \in \mathscr{F}_k^*(a,b)$ let $\psi(T,P) = (\mathscr{F}, T_0, T)$ where $\mathscr{F}$ is the Steel decomposition of T according to vertex $v_0 \in P$ and T_0 is the kernel of the decomposition. Since the Steel decomposition is unique and P is connected, the map ψ is well defined. If $\psi(T,P) = \psi(T',P')$ then $T = T'$ by the definition of ψ. The kernels of the decompositions are identical. Therefore $P = P'$, since both of them are an element of $\mu(T)$ which is in the kernel. So ψ is injective. Finally, Lemma 4 proves that ψ is onto. □

Lemma 7.

$$f_k(a,b) = (k-1)!(2n-3k)N(a,k)N(b,k)\frac{b(n)}{b(n-k+2)}.$$

Proof. We know that $|\mathscr{F}_k(a,b)| = f_k(a,b)$. Therefore $|\mathscr{F}_k^*(a,b)| = kf_k(a,b)$. Now we have

$$\begin{aligned}
|\mathscr{G}_k(a,b)| &= \sum_{\mathscr{F} \in \mathscr{H}(a,b,k)} \sum_{T_0 \in \mathscr{F}} |\mathrm{Ext}(T_0; \mathscr{F} \setminus \{T_0\})| \\
&= \sum_{\mathscr{F} \in \mathscr{H}(a,b,k)} \sum_{T_0 \in \mathscr{F}} e_0 \frac{b(n)}{b(n-k+2)} \\
&= (2n-3k)|\mathscr{H}(a,b,k)|\frac{b(n)}{b(n-k+2)}.
\end{aligned}$$

Furthermore, we know that $|\mathscr{H}(a,b,k)| = k!N(a,k)N(b,k)$. (The forests of $\mathscr{H}(a,b,k)$ can be built as follows: take a semilabelled forest of k rooted binary trees with leaf set L_A and a semilabelled forest of k rooted binary trees with leaf set L_B, match them up and make bichromatic rooted binary trees from the pairs.) Now Lemma 6 finishes the proof. □

References

[1] M. Carter, M. Hendy, D. Penny, L.A. Székely and N.C. Wormald, On the distribution of lengths of evolutionary trees, SIAM J. Discrete Math. 3 (1990) 38–47.
[2] P.L. Erdős, A new bijection on rooted forests, Discrete Math. 111 (1993) 179–188.
[3] P.L. Erdős and L.A. Székely, Application of antilexicographic order I, An enumerative theory of trees, Adv. Appl. Math. 10 (1989) 488–496.
[4] J. Felsenstein, Phylogenies from molecular sequences: Inference and reliability, Ann. Rev. Genetics 22 (1988) 521–565.
[5] W.M. Fitch, Towards defining the course of evolution: Minimum change for specific tree topology, Systems Zool. 20 (1971) 406–416.
[6] R.L. Graham and L.R. Foulds, Unlikelihood that minimal phylogenies for a realistic biological study can be constructed in reasonable computational time, Math. Biosci. 60 (1982) 133–142.
[7] J.A. Hartigan, Minimum mutation fits to a given tree, Biometrics 29 (1973) 53–65.
[8] K. Menger, Zur allgemeinen Kurventheorie, Fund. Math. 10 (1926) 96 115.
[9] J.W. Moon, Counting Labelled Trees, Canadian Mathematical Congress, Montreal, Que. (1970).
[10] M.A. Steel, Distributions on bicoloured binary trees arising from the principle of parsimony, Discrete Appl. Math. 41 (1993) 245–261.

Mathematical Programming 65 (1994) 93–105

On weighted multiway cuts in trees

Péter L. Erdős*,[a], László A. Székely**,[b]

[a]*Centrum voor Wiskunde en Informatica, 1098 SJ Amsterdam, Netherlands*
Mathematical Institute of the Hungarian Academy of Sciences, H-1055 Budapest, Hungary
[b]*Department of Computer Science, Eötvös University, H-1088 Budapest, Hungary*
Department of Mathematics, University of New Mexico, Albuquerque, NM 87131, USA

Received 11 September 1991; revised manuscript received 1 April 1993

Abstract

A min–max theorem is developed for the multiway cut problem of edge-weighted trees. We present a polynomial time algorithm to construct an optimal dual solution, if edge weights come in unary representation. Applications to biology also require some more complex edge weights. We describe a dynamic programming type algorithm for this more general problem from biology and show that our min–max theorem does not apply to it.

AMS 1991 Subject Classifications: 05C05, 05C70, 90C27

Keywords: Multiway cut; Menger's theorem; Tree; Duality in linear programming; Dynamic programming

1. Introduction

Let $G=(V, E)$ be a simple graph, $C=\{1, 2, \ldots, r\}$ be a set of colours. For $N \subseteq V(G)$, a map $\chi: N \to C$ is a *partial colouration*. We usually think of a given partial colouration. A map $\bar{\chi}: V(G) \to C$ is a *colouration* if $\chi(v) = \bar{\chi}(v)$ holds for all $v \in N$.

A *colour dependent weight function* assigns to every edge (p, q) and colours i, j a natural number $w(p, q; i, j)$, which tells the weight of the edge (p, q) in a colouration $\bar{\chi}$, in which $\bar{\chi}(p)=i$, $\bar{\chi}(q)=j$. We assume that $w(p, q; i, i)=0$ and $w(p, q; i, j)=w(q, p; j, i)$. We say that w is *colour independent*, if for any (p, q), $i_1 \neq j_1$, $i_2 \neq j_2$, we have $w(p, q; i_1, j_1)=w(p, q; i_2, j_2)$. We say that w is *edge independent*, if for any $(p_1, q_1) \in E$ and $(p_2, q_2) \in E$, and

*Corresponding author.

**Research of the author was supported by the A. v. Humboldt-Stiftung and the U.S. Office of Naval Research under the contract N-0014-91-J-1385.

SSDI 0025-5610(93)E0073-N

$i, j \in C$, we have $w(p_1, q_1; i, j) = w(p_2, q_2; i, j)$. (Hence, any edge independent weight function satisfies $w(p, q; i, j) = w(p, q; j, i)$.) We say that w is *constant*, if it is colour and edge independent.

An edge (p, q) is *colour-changing* in the colouration $\bar{\chi}$, if $\bar{\chi}(p) \neq \bar{\chi}(q)$. The *changing number* of the colouration $\bar{\chi}$ is the sum of weights of the colour-changing edges in $\bar{\chi}$, i.e.:

$$\textbf{change}(G, \bar{\chi}) = \sum_{(p, q) \in E(G)} w(p, q; \bar{\chi}(p), \bar{\chi}(q)) .$$

A partial colouration χ defines a partition of N by $N_i = \{v \in N: \chi(v) = i\}$. A set of edges that separates every N_i from all the other N_j's is termed a *multiway cut* [1]. Observe that the set of colour-changing edges of a colouration $\bar{\chi}$ forms a multiway cut and every multiway cut is represented in this way.

The *length* of the pair (G, χ) is the minimum weight of a multiway cut, in formula:

$$l(G, \chi) = \min\{\textbf{change}(G, \bar{\chi}): \bar{\chi} \text{ colouration}\} .$$

An *optimal colouration* is a colouration $\bar{\chi}$ such that $\textbf{change}(G, \bar{\chi}) = l(G, \chi)$.

The multiway cut problem for colour independent weight functions has been extensively studied in combinatorial optimization (e.g. [1–3].). As Dahlhaus et al. pointed out [3], this problem is NP-hard, even for $|N| = 3$, $|N_i| = 1$ and constant weight.

On the other hand, if we restrict ourselves to planar graphs, a fixed number of colours, and constant weight, then the problem becomes solvable in polynomial time [3]. A well-known specialization of the multiway cut problem, which is solvable in polynomial time, is $r = 2$, which is considered in the undirected edge version of Menger's theorem [8].

Although it is less known in the operations research community, some instances of the multiway cut problem have great importance in biomathematics. In fact, the notions of the changing number and the length came from genetics and we follow the terminology used there. For the case of constant weight function, Fitch [6] and Hartigan [7] developed a polynomial time algorithm to determine the length of a given tree. Sankoff and Cedergren [13], and Williamson and Fitch [12] studied edge independent weight functions and made polynomial time algorithms to find the length. Some explanation of the significance of the multiway cut problem in biology is given in [4, 5].

The goal of the present paper is to study the multiway cut problem. In Section 2 we give a new lower bound for the length of a multiway cut. Section 3 provides a dynamic programming type algorithm to find the length of a tree with an arbitrary weight function. Section 4 uses the algorithm of Section 3 to establish a min–max theorem for the multiway cut problem of trees, in the case of colour independent weight functions. All the results can be extended to any graph G, in which N intersects every cycle. Section 5 describes our results in terms of linear programming.

A preliminary version of the present paper has already appeared [5]. We are indebted to the anonymous referees for their helpful observations that we use in this presentation.

2. Lower bound for the weight of a multiway cut

Let G be a simple graph, $N \subseteq V(G)$ and $\chi: N \to C$ be a partial colouration. Let w be a colour dependent weight function.

Definition. An oriented path P in G starting at $s(P) \in N$ and terminating at $t(P) \in N$ is a *colour-changing path*, if $\chi(s(P)) \neq \chi(t(P))$ and P has no internal vertex in N. (From now on path means oriented path, unless we explicitly say the opposite.) Let us fix a family $\mathscr{P}$ of colour-changing paths and let $e = (p, q) \in E(G)$. Define

$$n_i(e, \mathscr{P}) = \#\{P \in \mathscr{P}: (p, q) \in P \text{ and } \chi(t(P)) = i\} .$$

The notation $(p, q) \in P$ means that P enters the edge (p, q) at p and leaves at q.

Definition. Let $\chi: N \to C$ be a partial colouration and $\bar{\chi}$ be a colouration on G. A family $\mathscr{P}$ of colour-changing paths is a *path packing*, if all pairs of colours $i \neq j$ and all edges (p, q) satisfy

$$n_i((p, q), \mathscr{P}) + n_j((q, p), \mathscr{P}) \leqslant w(p, q; j, i) .$$

The maximum cardinality of a path packing is denoted by $p(G, \chi)$.

Theorem 1. *For any graph G and partial colouration χ, we have*

$$l(G, \chi) \geqslant p(G, \chi) .$$

Proof. Let $\mathscr{P}$ be a path packing and $\bar{\chi}: V(G) \to C$ be an optimal colouration. Define a map $f: \mathscr{P} \to E(G)$ as follows: let $f(P) = e$ if e is the last colour-changing edge in P in $\bar{\chi}$. For any colour changing edge $e = (p, q)$, $\bar{\chi}(p) = j$ and $\bar{\chi}(q) = i$ ($i \neq j$ since e is colour changing), we have

$$\#\{P \in \mathscr{P}: f(P) = e\} \leqslant n_i((p, q), \mathscr{P}) + n_j((q, p), \mathscr{P}) \leqslant w(p, q; j, i) .$$

Therefore,

$$|\mathscr{P}| \leqslant \mathbf{change}(G, \bar{\chi}) = l(G, \chi) . \qquad \square$$

3. An algorithm to find optimal colourations

Now we focus on the multiway cut problem of trees. Let T be a tree and $\chi: N \to C$ be a partial colouration, and let $L(T)$ denote the set of leaves, i.e. vertices of degree 1. We assume $N = L(T)$. (It is obvious that the solution of the multiway cut problem of trees with $N = L(T)$ easily generalizes to the solution of the multiway cut problem of trees with arbitrary N.) Let w be a colour dependent weight function. In this section we give a polynomial time algorithm to determine all optimal colouration of T for the weight w.

Let us fix an arbitrary non-leaf vertex, the **root** of T. Let (u, v) be an edge and let v be closer to the **root** than u, then we say $v = \mathbf{Father}(u)$. (**Father(root)** is **NIL.**) We denote the set of all u for which $v = \mathbf{Father}(u)$ by $\mathbf{Son}(v)$.

Our colouring algorithm has two phases. Starting from the leaves and approaching the **root** we determine a *penalty function* of every vertex v recursively, and subsequently we determine a suitable colouration $\bar{\chi}$ starting from the *root* and spreading to the leaves.

Definition. The vector-valued *penalty function* is a map

$$\mathbf{pen}: V(T) \to (\mathbb{N} \cup \{\infty\})^r,$$

such that $\mathbf{pen}_i(v)$ means the length of the subtree separated by v from the **root**, if the colour of v has to be i.

Phase I. For every leaf $v \in L(T)$ let

$$\mathbf{pen}_i(v) = \begin{cases} 0 & \text{if } v \in N_i, \\ \infty & \text{otherwise}, \end{cases}$$

where in an actual computation ∞ may be substituted by a sufficiently large number. Take a vertex v, such that $\mathbf{pen}(v)$ is not computed yet for the vertex v, but $\mathbf{pen}(u)$ is already known for every vertex $u \in \mathbf{Son}(v)$. Then compute

$$\mathbf{pen}_i(v) = \sum_{u \in \mathbf{Son}(v)} \min_{j=1, \ldots, r} \{w(u, v; j, i) + \mathbf{pen}_j(u)\}.$$

Phase II. Now we determine an optimal colouration $\bar{\chi}$ of T. First, let $\bar{\chi}(\mathbf{root})$ be a colour i, which minimizes the value $\mathbf{pen}_i(\mathbf{root})$. Furthermore, for a vertex v for which $\bar{\chi}(v)$ is not settled yet, but $\bar{\chi}(\mathbf{Father}(v))$ is already determined, let $\bar{\chi}(v)$ be a colour i, which minimizes the expression

$$w(v, \mathbf{Father}(v); i, \bar{\chi}(\mathbf{Father}(v))) + \mathbf{pen}_i(v).$$

It is easy to see, that every leaf $v \in N_i$ satisfies $\bar{\chi}(v) = i = \chi(v)$, for $i = 1, \ldots, r$.

The correctness of this algorithm is almost self-explanatory. Assume the positive integer edge weights are given in unary representation. Then, the time complexity is $O(n \cdot r^2 \cdot (\max \textit{weight}))$, since at each step we calculate r^2 sums, take the minimum, and roughly $2n$ steps are necessary because T has n vertices and $n-1$ edges. You may change max *weight* for $\log(\max \textit{weight})$, if the edge weights come in binary representation.

In the rest of this section we focus on colour independent weight functions, since we can develop a slightly more efficient version of this algorithm, which also can determine all optimal colourations. Biologists may need all optimal colourations; the saving in running time comes from avoiding the second minimization in Phase II. Also, case (A2) in the proof of Theorem 2 will need the modified algorithm. For the sake of simplicity, for the rest of this section the weight function is a map $w: E(T) \to \mathbb{N}$ for colour changing edges

and the weight of any edge not changing colour is 0. We use the usual *Kronecker delta* notation.

Phase I′. For every leaf v, set

$$M_1(v) = M_2(v) = \{i\colon \mathbf{pen}_i(v) = 0\}\ .$$

If $\mathbf{pen}(v)$ is not computed yet for the vertex v but $\mathbf{pen}(u)$ is already known for every vertex $u \in \mathbf{Son}(v)$, then set

$$\mathbf{pen}_i(v) = \sum_{u \in \mathbf{Son}(v)} \min_{j=1,\ i,\ r} \{(1-\delta_{ij})w(u, v) + \mathbf{pen}_j(u)\}\ .$$

Let $p(v) = \min_i \mathbf{pen}_i(v)$, and

$$M_1(v) = \{i \in \{1, \ldots, r\}\colon \mathbf{pen}_i(v) = p(v)\}\ ,$$

$$M_2(v) = \{i \in \{1, \ldots, r\}\colon \mathbf{pen}_i(v) < p(v) + w(v, \mathbf{Father}(v))\}\ .$$

It is obvious that $M_1(v) \subseteq M_2(v)$.

Phase II′. For $\bar{\chi}(\mathbf{root})$, take an arbitrary element of $M_1(\mathbf{root})$. If $\bar{\chi}(v)$ is not settled yet for a vertex v, but $\bar{\chi}(\mathbf{Father}(v))$ is already determined, take

$$\bar{\chi}(v) = \begin{cases} \bar{\chi}(\mathbf{Father}(v)) & \text{if } \bar{\chi}(\mathbf{Father}(v)) \in M_2(v)\ , \\ \text{an arbitrary element of } M_1(v) & \text{otherwise}\ . \end{cases}$$

It is easy to see, that every vertex $v \in N_i$ satisfies $\bar{\chi}(v) = i = \chi(v)$, for $i = 1, \ldots, r$. This algorithm is obviously correct and permitting some extra freedom at certain steps, any optimal colouration can be obtained by the modified algorithm. For this purpose we introduce a third set of colours at Phase I′:

$$M_3(v) = \{i \in \{1, \ldots, r\}\colon \mathbf{pen}_i(v) = p(v) + w(v, \mathbf{Father}(v))\}\ .$$

If in Phase II′ we also allow to give the colour of $\bar{\chi}(\mathbf{Father}(v))$ to v, if $\bar{\chi}(\mathbf{Father}(v)) \in M_3(v)$, then the algorithm still yields an optimal colouration. Moreover, one can prove that running this algorithm in all possible ways yields all optimal colourations. (We leave the proof to the reader.) The complexity of this revised algorithm is better by a constant multiplicative factor than that of the original, but to get every optimal colouration may take exponential time, since M.A. Steel exhibited trees with exponentially many optimal colourations [11].

4. A min–max theorem

In this section we assume that the weight function is *colour-independent* and we prove that the lower bound of Theorem 1 is tight for leaf-coloured trees, and then even for a larger class of graphs.

Theorem 2. *Let T be an arbitrary tree with colour-independent weight function $w: E(T) \to \mathbb{N}$ and with leaf-colouration $\chi: L(T) \to C$. Then*

$$l(T, \chi) = p(T, \chi) .$$

We already know from Theorem 1 that the LHS is greater or equal than the RHS. We have to prove the other inequality. For this end we construct the desired optimal path packing in a recursive manner. At first, we explicitly construct optimal path packings for stars, i.e. for trees with 1 branching vertex. Then, for a tree T with at least 2 branching vertices and with

$$W(T) = \sum_{f \in E(T)} w(f)$$

sum of weights, we define a 'smaller' tree T' for which we can trace back the problem of the construction of an optimal path packing, such that we can 'lift up' the path packing from T' to T to get the solution. We may have at most $W(T)$ 'lift up' steps. Here we give the details.

For convenience, we want to use the functions **Son** and **Father**, therefore we fix, as in Section 3, a **root** of T. In the complexity issues we assume that our tree is represented by the vertices v and the sets **Son**(v) and **Father**(v), furthermore every element of **Son**(v) and **Father**(v) (which represents edges) also contains the weight of the edge. The paths under construction will be represented as double-linked lists, therefore, due to Theorem 1, the space complexity of the representation is $O(l(T, \chi) \cdot n)$.

Definition. We say that a vertex v is *of order* 1 if every element of **Son**(v) is a leaf.

Notice that every tree with at least 2 branching vertices has a non-root vertex of order 1. Before starting the main body of the proof we need the following lemma.

Lemma 1. *One can assume that no vertex of order* 1 *has two sons with the same colour.*

Let v be a vertex of order 1, such that **Son**(v) contains at least 2 leaves with identical colour. Let $\Sigma(T)$ denote the tree obtained from T by identification of the elements of **Son**(v) with identical colour and adding up their edge weights, respectively. Now one can easily construct an optimal path packing for T from an optimal path packing of $\Sigma(T)$. Anyhow, we give a formal proof, otherwise, the base case of our recursive algorithm would not be complete.

Proof. Define the tree $\Sigma(T)$ formally as follows: let the tree T' be a star with midpoint v and with leaves $\{l_i: \exists u \in \mathbf{Son}(v)$ with $\chi(u) = i\}$ and let $\Sigma(T)$ be the tree made of the trees $T \setminus \mathbf{Son}(v)$ and T' by identification of their common v. The leaf-colouration and weight function of $\Sigma(T)$ are as follows:

$$\chi'(u) = \begin{cases} \chi(u) & \text{if } u \in L \setminus \mathbf{Son}(v) , \\ i & \text{if } u = l_i , \end{cases}$$

$$w'(f) = \begin{cases} \sum_{\substack{u \in \mathbf{Son}(v) \\ \chi(u) = i}} w((u, v)) & \text{if } f = (l_i, v)\,, \\ w(f) & \text{otherwise}\,. \end{cases}$$

Notice that $l(\Sigma(T), \chi') = l(T, \chi)$.

Claim. *If* $l(\Sigma(T), \chi') = p(\Sigma(T), \chi')$ *then* $l(T, \chi) = p(T, \chi)$.

Proof. Let $\mathbf{Son}(v)$ contain d different colours. We apply induction on $|\mathbf{Son}(v)|$.

Base case: if $|\mathbf{Son}(v)| = d$, then $\Sigma(T) = T$, $\chi = \chi'$, and we have nothing to prove.

Inductive step: Suppose that we know Lemma 1 for all $|\mathbf{Son}(v)| < k$. Assume now $|\mathbf{Son}(v)| = k$ and for some fixed $z_1, z_2 \in \mathbf{Son}(v)$, let $\chi(z_1) = \chi(z_2)$. Join z_1 and z_2 into z. In the new tree T^* obtained by identification, define the leaf colouration and the weight function as follows:

$$\chi^*(u) = \begin{cases} \chi(u) & \text{if } u \neq z_1, z_2\,, \\ \chi(z_1) & \text{if } u = z\,, \end{cases}$$

$$w^*(f) = \begin{cases} w(f) & \text{if } f \neq (v, z_i)\,, \\ w(v, z_1) + w(v, z_2) & \text{if } f = (v, z)\,. \end{cases}$$

Now we have $\Sigma(T) = \Sigma(T^*)$, therefore $l(\Sigma(T)) = l(\Sigma(T^*))$. By the hypothesis there exists a path packing $\mathscr{P}^*$ in the tree T^* satisfying $|\mathscr{P}^*| = l(T^*)$. It is easy to divide the paths of $\mathscr{P}^*$ adjacent to vertex z into two groups, such that the members of one group are adjacent to z_1 and the members of the other are adjacent to z_2 and both groups obey the weight restriction on the edge adjacent to z_i. In this way we obtain a path packing of $l(T)$ members in T. This proves the Claim as well as Lemma 1. □

The time complexity of this algorithm is $O(\sum_{u \in \mathbf{Son}(v)} w(u, v))$ so the time complexity of all applications of Lemma 1 altogether is $O(W(T))$.

We return to the main body of the proof; we assume that any two sons of an arbitrary vertex of order 1 have different colours. Our algorithm is given in a recursive form in the variables $b(T)$ and $W(T)$, where $b(T)$ is the number of branching (non-leaf) vertices of T.

Base case: let $b(T) = 1$ and $W(T)$ be arbitrary. Then T is a star; let v denote the midpoint of it. Due to Lemma 1 we may assume that $|L(T)| = r$ (i.e. every colour occurs once). Assume that the edge (v, u) has maximum weight over all edges. Orient paths from u to every other leaf $z \in L(T) \setminus \{u\}$ with multiplicity $w(v, z)$. This path system is obviously a path packing and has $l(T)$ members. This case requires $O(W(T))$ steps.

Recursive step: For any tree T with at least 2 branching vertices we shall find 'smaller' tree T' with fewer branching vertices ($b(T') < b(T)$) or with smaller total weights

($b(T')=b(T)$ and $W(T')<W(T)$) such that an optimal path packing of T' can be lifted up to an optimal path packing of T. Define

$$s(v)=\max_{u\in\mathbf{Son}(v)} w(v,u)\ .$$

We distinguish two cases:

(A) There is a vertex v of order 1 such that $s(v)\neq w(v,\mathbf{Father}(v))$.

(B) $s(v)=w(v,\mathbf{Father}(v))$ for every vertex v of order 1.

Case (A). Let $\bar{\chi}$ be an optimal colouration of T such that v is the first branching vertex for which the colour sets M_i were determined. We have two subcases; in (A1) we have $s(v)>w(v,\mathbf{Father}(v))$, in (A2) we have $s(v)<w(v,\mathbf{Father}(v))$.

Case (A1). Let T'' be the tree with the same vertex set, edge set and leaf colouration as the tree T was, and let the new weight function $w':E(T)\to\mathbb{N}$ such that

$$w'(f)=\begin{cases} w(f)-1 & \text{if } f=(v,u) \text{ where } u\in\mathbf{Son}(v)\ ,\\ w(f) & \text{if otherwise}\ .\end{cases}$$

If $w'(f)=0$, then cancel this edge and its leaf endpoint from the tree T'' to obtain the tree T'. Due to our colouring algorithm, colouration $\bar{\chi}$ is also optimal for the tree T', therefore

$$l(T')+(|\mathbf{Son}(v)|-1)=l(T)\ .$$

The total weight of tree T' is less than of T. Assume now that we have an optimal path packing $\mathscr{P}'$ of $l(T',\chi)$ elements in T'. Denote by ΔT the star of $v\cup\mathbf{Son}(v)$ with weight function $w\equiv 1$ and with the original leaf colouration. Let $\Delta\mathscr{P}$ be optimal path packing in ΔT (use the base case). Now the path system $\mathscr{P}=\mathscr{P}'\cup\Delta\mathscr{P}$ is obviously optimal path packing in the tree T.

We can construct T' and the path packings $\Delta\mathscr{P}$ and $\mathscr{P}$ from the given tree T and path packing $\mathscr{P}'$ in $O(r\cdot\sum_{u\in\mathbf{Son}(v)} w(v,u))$ time, so that the total time complexity of the case (A1) is $O(rW(T))$.

Case (A2). Now we have $s(v)<w(v,\mathbf{Father}(v))$. Let the tree T' be identical with the tree T with the same leaf-colouration and with the weight function

$$w'(f)=\begin{cases} s(v) & \text{if } f=(v,\mathbf{Father}(v))\ ,\\ w(f) & \text{otherwise}\ .\end{cases}$$

Now it is easy to see that there exists an optimal colouration $\bar{\chi}$ of T' satisfying $\bar{\chi}(v)=\bar{\chi}(\mathbf{Father}(v))$ which is also optimal in T. (The only problem that can occur is that $\bar{\chi}(\mathbf{Father}(v))\in M_2(v)$ but $\bar{\chi}(\mathbf{Father}(v))\in M_3'(v)$. In that case we can apply the extended Phase II'.) Therefore, we have $l(T)=l(T')$ and $W(T')<W(T)$. Now we can easily 'lift up' any optimal path packing $\mathscr{P}$ of T' to the tree T, namely $\mathscr{P}$ itself is obviously path packing in T.

This operation takes $O(1)$ time, so the total time complexity of case (A2) is $O(n)$.

Case (B). From now on we assume that every vertex z of order 1 satisfies the condition $s(z)=w(z,\mathbf{Father}(z))$. For the rest of (B), we fix a vertex v; if the diameter of T is 3, then

let v be the **root**, otherwise, let v be a non-root vertex such that $\mathbf{Son}(v) \not\subset L(T)$ and every non-leaf son is a vertex of order 1 (the existence of such a v is obvious). Let the non-leaf sons of v be the vertices $z_1, \ldots, z_k$.

By the definition of case (B) it is easy to see the existence of an optimal coloration $\bar{\chi}$ colouring v and every z_i to the same colour. Therefore if $\bar{T}$ is the tree derived from the tree T by contracting every edge of form (v, z_i) (leaving the name of the new vertex v), which is endowed with the original leaf-colouration and weight function on the existing edges, then the restriction of the same colouration $\bar{\chi}$ is also optimal for $\bar{T}$ and $l(\bar{T}) = l(T)$. On the other hand, the tree $\bar{T}$ has less branching vertices than T.

Now due to our hypothesis we have an optimal path packing $\bar{\mathscr{P}}$ in the tree $\bar{T}$. Therefore

$$|\bar{\mathscr{P}}| = l(T) .$$

Let us define the *lift up* $\mathscr{P} = \{\hat{P}: P \in \bar{\mathscr{P}}\}$ of the path packing $\bar{\mathscr{P}}$, where $\hat{P}$ is identical with P if no leaf u of $\mathbf{Son}(z_i)$ $(i = 1, \ldots, k)$ belongs to the path P, and $\hat{P}$ comes from P by subdivision of the edge (v, u) with vertex z_i if $\text{endvertex}(P) = u \in \mathbf{Son}(z_i)$ $(i = 1, \ldots, k)$. We have $l(T)$ many elements in $\mathscr{P}$.

Let $e_i = (v, z_i)$ (for every $i = 1, \ldots, k$). For an edge $f = (p, q)$, we write $-f = (q, p)$. Now, by the definition of $\mathscr{P}$, the condition

$$n_i(f, \mathscr{P}) + n_j(-f, \mathscr{P}) \leqslant w(f)$$

holds for every edge $f \neq e_i$ $(i = 1, \ldots, k)$, but unfortunately this is not necessarily the case for the edges e_i.

We solve this problem in a slightly more general setting (Lemma 2). For this we introduce the following notations: Let $[x]^+$ denote x, if x is non-negative, 0, if x is non-positive. Define the *badness* of the colour changing path system $\mathscr{P}$ by

$$\text{bad}(\mathscr{P}) = \sum_{\substack{(i,j) \in C \times C \\ i \neq j}} \sum_{e \in E(G)} [n_i(e, \mathscr{P}) + n_j(-e, \mathscr{P}) - w(e)]^+ .$$

Call an edge *oversaturated* by the path system $\mathscr{P}$, if the contribution of the edge to the badness is positive. (We recall the definition $e_i = (v, z_i)$.)

Lemma 2. *Let $\mathscr{P}$ be a system of colour-changing paths on the tree T such that*

(i) *for all i, j, $n_j(\pm e_i, \mathscr{P}) \leqslant w(e_i)$,*

(ii) *$\mathscr{P}$ does not oversaturate any edge from $E(T) \setminus \{e_1, \ldots, e_k\}$.*

Then there exists a path packing $\mathscr{P}^*$ in T of the same size.

Proof. If $\text{bad}(\mathscr{P}) = 0$ then $\mathscr{P}$ itself is a path packing. Suppose $\text{bad}(\mathscr{P}) > 0$, and, say, the edge e_1 is oversaturated with colours 1 and 2, i.e.

$$n_1(e_1, \mathscr{P}) + n_2(-e_1, \mathscr{P}) > w(e_1) .$$

Take a path $P_1 \in \mathscr{P}$ such that $e_1 \in P_1$ and $\chi(t(P_1)) = 1$ (where, say, $t(P_1) \in \mathbf{Son}(z_1)$), and a path $P_2 \in \mathscr{P}$ such that $-e_1 \in P_2$ and $\chi(t(P_2)) = 2$ (where $t(P_2) \notin \mathbf{Son}(z_1)$ and $s(P_2) \in \mathbf{Son}(z_1)$). Now we distinguish the cases (BA) and (BB):

Case (BA). Suppose there is no $P_3 \in \mathscr{P}$ for which $-e_1 \in P_3$, $s(P_3) = s(P_2)$ and $\chi(t(P_3)) = 1$. In this case we define the following path system:

$$\mathscr{P}_1 = \mathscr{P} \cup \{P\} \setminus \{P_1\} ,$$

where the path P is $(s(P_2), z_1, t(P_1))$, oriented from left to right.

Claim A.

$$\mathrm{bad}(\mathscr{P}_1) \leqslant \mathrm{bad}(\mathscr{P}) - 1 .$$

Proof. It is easy to see that $n_i(\pm f, \mathscr{P}_1) \leqslant n_i(\pm f, \mathscr{P})$ for each $i = 1, \ldots, k$ and for each $f \in E(T) \setminus \{e_1, (z_1, s(P_2))\}$, furthermore

$$n_i(-e_1, \mathscr{P}_1) = n_i(-e_1, \mathscr{P}), \quad i = 1, \ldots, k ,$$

$$n_i(e_1, \mathscr{P}_1) = n_i(e_i, \mathscr{P}), \quad i = 2, \ldots, k ,$$

$$n_1(e_1, \mathscr{P}_1) = n_1(e_1, \mathscr{P}) - 1 .$$

Finally, for the edge $f_2 = (z_1, s(P_2))$ we have

$$n_i(f_2, \mathscr{P}_1) = n_i(f_2, \mathscr{P}), \quad i = 1, \ldots, k ,$$

$$n_i(-f_2, \mathscr{P}_1) = n_i(-f_2, \mathscr{P}), \quad i = 2, \ldots, k ,$$

$$n_1(-f_2, \mathscr{P}_1) + n_i(f_2, \mathscr{P}_1) \leqslant w(f_2), \quad i = 1, \ldots, k .$$

The last inequality is true, since otherwise $n_2(-f_2, \mathscr{P}) + n_i(f_2\ \mathscr{P}) > w(f_2)$ would hold, contradicting the assumptions of Lemma 2. □

Case (BB). Suppose there exists a path P_3 which was forbidden in (BA). Then let $\mathscr{P}_1$ be the following path system:

$$\mathscr{P}_1 = \mathscr{P} \cup \{P, P_3 \wedge P_1\} \setminus \{P_1, P_3\}$$

where $P_3 \wedge P_1$ denotes the (unique) path oriented from $s(P_3)$ to $t(P_1)$.

Claim B.

$$\mathrm{bad}(\mathscr{P}_1) \leqslant \mathrm{bad}(\mathscr{P}) - 1 .$$

Proof. Set

$$E_1 = \{e_1, (z_1, t(P_1)), (z_1, s(P_3))\} \quad \text{and} \quad E_2 = E(P_1) \cup E(P_2) \setminus E(P_3 \wedge P_1).$$

Then for each edge $f \in E(T) \setminus (E_1 \cup E_2)$ the estimates of Claim A hold. Furthermore, for $f \in E_1$ we have

$$n_i(\pm f, \mathscr{P}_1) = n_i(\pm f, \mathscr{P}), \quad i = 2, \ldots, k,$$

$$n_1(\pm f, \mathscr{P}_1) \leqslant n_1(\pm f, \mathscr{P}),$$

$$n_i(\pm(z_1, t(P_1)), \mathscr{P}_1) = n_i(\pm(z_1, t(P_1)), \mathscr{P}), \quad i = 1, \ldots, k,$$

$$n_i(\pm e_1, \mathscr{P}_1) = n_i(\pm e_1, \mathscr{P}), \quad i = 2, \ldots, k,$$

$$n_1(\pm e_1, \mathscr{P}_1) = n_1(\pm e_1, \mathscr{P}) - 1,$$

$$n_i(\pm(z_1, s(P_3)) = n_i(\pm(z_1, s(P_3)), \mathscr{P}) \quad i = 1, \ldots, k.$$

The equalities and inequalities above prove Claim B. □

The surgeries described in Case (BA) and Case (BB) obviously keep the conditions of Lemma 2, therefore they may be repeated until the badness drops to 0. Claims A and B guarantee, that we finally reach 0. Lemma 2 and Theorem 2 are proved. □

The determination of the tree $\bar{T}$ takes $O(n)$ steps, therefore the total time complexity of this procedure is $O(nb(T))$. To lift up the paths from $\bar{\mathscr{P}}$ to $\mathscr{P}$ takes

$$O\left(r \sum_{z \in \mathbf{Son}(v)} w(v, z)\right)$$

time, therefore the total time complexity of lift up operations is $O(rW(T))$. Finally, the badness at Lemma 2 is at most

$$\sum_{z \in \mathbf{Son}(v)} w(v, z)$$

and every edge can occur at most one application of Lemma 2 so the total time complexity of Lemma 2 is $O(\max\{rW(T), n^2\})$.

The bookkeeping of (edge, path) incidences is necessary. A possible execution of this task is to build up lists for every edge to store these incidences and to maintain these lists at every 'lift up' step. The total time complexity of our recursive procedure is $O(\max\{rW(T), n^2\})$, so it is unary polynomial.

The following theorem is an easy consequence of Theorem 2.

Theorem 3. *Let G be a graph with a weight function $w: E(T) \to \mathbb{N}$ and with a partial colouration $\chi: N \to C$. Assume that N intersects every cycle of G. Then*

$$l(G, \chi) = p(G, \chi)$$

Proof. Obtain a forest by eliminating the vertices of N and making leaves from the edges that were adjacent to them. Give the colour of n to the leaves that substitute a former $n \in N$. Apply Theorem 2 for each and every tree in the forest. □

5. The LP connection

One may consider the following linear programs related to the multiway cut problem with colour independent weight function. Note that this is something, which is different from the usual multiway cut polyhedron [1].

For every oriented edge (p, q) of G and every ordered pair of distinct colours ij define a variable $z_{pq,ij}$. If $q \in N$, then eliminate $z_{pq,ij}$ and $z_{qp,ji}$ for every $j \neq \chi(q)$. Introduce new quotient variables by identifying the surviving variables $z_{pq,ij}$ and $z_{qp,ji}$ in pairs. For convenience we use the same notation for the quotient variables. Then the primal linear program is:

$$z_{pq,ij} \geqslant 0\,;$$

for every colour-changing path P_{ab} $(a, b \in N)$, have

$$\sum_{(p,\,q)\in P_{ab}} \sum_{i:\,i\neq\chi(b)} z_{pq,i\chi(b)} \geqslant 1;$$

$$\min \sum z_{pq,ij}\, w(p, q)\,,$$

where the last sum is for all quotient variables. To describe the dual linear program, for every colour-changing path P_{ab} introduce a variable λ_{ab}, such that

$$\lambda_{ab} \geqslant 0\,;$$

for every quotient variable $z_{pq,ij}$, have

$$\sum_{\substack{\chi(b)=j \\ (p,\,q)\in P_{ab}}} \lambda_{ab} + \sum_{\substack{\chi(v)=i \\ (q,\,p)\in P_{uv}}} \lambda_{uv} \leqslant w(p, q);$$

$$\max \sum \lambda_{ab}\,.$$

We claim that these linear programs have integer optimal solutions. It is easy to see, that

$$p(G, \chi) \leqslant \max \sum \lambda_{ab} : \lambda_{ab} \text{ integer} \leqslant \max \sum \lambda_{ab} = \min \sum z_{pq,ij}\, w(p, q)$$

$$\leqslant \min \sum z_{pq,ij}\, w(p, q) : z_{pq,ij} \text{ integer} \leqslant l(G, \chi)\,.$$

Only the first and last inequalities require proofs from the chain of inequalities above. The first one holds, since any path packing provides a feasible integer solution for the second linear program. The last one holds, since we have an optimal colouration $\bar{\chi}$ with total weight

of the colour-changing edges of $l(G, \chi)$; define $z_{pq,ij}=1$, iff (p, q) is a colour-changing edge in the optimal colouration $\bar{\chi}$ and $\bar{\chi}(p)=i$, $\bar{\chi}(q)=j$ hold, and $z_{pq,ij}=0$ otherwise. If $l(G, \chi)=p(G, \chi)$. then equality holds everywhere in the chain.

It is a natural question whether these linear programs are totally dual integral [10], i.e., whether they have integer optimal solutions for colour dependent weight functions $w(p, q; i, j)$. Unfortunately, this is not the case, take for example the 3-star with center c and leaves x, y, z with colours $\chi(x)=1$, $\chi(y)=2$ and $\chi(z)=3$; and the weight function $w(c, .; i, j)={}_iW_j$ defined by the matrix

$$W=\begin{pmatrix}0 & 1 & 3\\ 3 & 0 & 1\\ 1 & 3 & 0\end{pmatrix}.$$

References

[1] S. Chopra and M.R. Rao, "On the multiway cut polyhedron," *Networks* 21 (1991) 51–89.

[2] W.H. Cunningham, "The optimal multiterminal cut problem," *DIMACS Series in Discrete Math.* 5 (1991) 105–120.

[3] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P. Seymour and M. Yannakakis, "The complexity of multiway cuts," extended abstract (1983).

[4] P.L. Erdős and L.A. Székely, "Evolutionary trees: an integer multicommodity max–flow-min–cut theorem," *Advances in Applied Mathematics* 13 (1992) 375–389.

[5] P.L. Erdős and L.A. Székely, "Algorithms and min–max theorems for certain multiway cut," in: E. Balas, G. Cornuéjols and R. Kannan, eds., *Integer Programming and Combinatorial Optimization*, Proceedings of the Conference held at Carnegie Mellon University, May 25–27, 1992, by the Mathematical Programming Society (CMU Press, Pittsburgh, 1992) 334–345.

[6] W.M. Fitch, "Towards defining the course of evolution. Minimum change for specific tree topology," *Systematic Zoology* 20 (1971) 406–416.

[7] J.A. Hartigan, "Minimum mutation fits to a given tree," *Biometrics* 29 (1973) 53–65.

[8] L. Lovász and M.D. Plummer, *Matching Theory* (North-Holland, Amsterdam, 1986).

[9] K. Menger, "Zur allgemeinen Kurventheorie," *Fundamenta Mathematicae 10 (1926) 96–115.*

[10] G.L. Nemhauser and L.A. Wolsey, *Integer and Combinatorial Optimization* (John Wiley & Sons, New York, 1988).

[11] M. Steel, "Decompositions of leaf-coloured binary trees," *Advances in Applied Mathematics* 14 (1993) 1–24.

[12] P.L. Williams and W.M. Fitch, "Finding the minimal change in a given tree," in: A. Dress and A. v. Haeseler, eds., *Trees and Hierarchical Structures*, Lecture Notes in Biomathematics 84 (1989) 75–91.

[13] D. Sankoff and R.J. Cedergren, "Simultaneous comparison of three or more sequences related by a tree," in: D. Sankoff and J.B. Kruskal, eds., *Time Wraps, String Edits and Macromoleculas: The Theory and Practice of Sequence Comparison* (Addison-Wesley, London, 1983) 253–263.

ELSEVIER

Discrete Applied Mathematics 87 (1998) 67–75

DISCRETE APPLIED MATHEMATICS

Minimum multiway cuts in trees

Péter L. Erdős [a,1], András Frank [b,c,2], László Székely [d,*,3]

[a] *Mathematical Institute of the Hungarian Academy of Sciences, Reáltanoda u. 13–15, Budapest H-1053, Hungary*

[b] *Department of Operations Research, Eötvös University, Múzeum krt. 6–8, Budapest H-1088, Hungary*

[c] *Ericsson Traffic Laboratory, Laborc u.1, Budapest H-1037, Hungary*

[d] *Department of Mathematics, University of South Carolina, Columbia, South Carolina, SC 29208, USA*

Received 1 May 1996; received in revised form 10 March 1998; accepted 30 March 1998

Abstract

We compare three lower bounds for the minimum cardinality of a multiway cut in a graph separating a given set S of terminals. The main result is a relatively short algorithmic proof for a simplified version of a min–max theorem of the first and the third authors asserting that the best of the three lower bounds is actually attainable if every circuit of the graph contains a terminal node.

0. Introduction

Let $G=(V,E)$ be a connected graph with no loops and S a specified subset of nodes. A family $\mathscr{P}:=\{V_1,V_2,\ldots,V_t\}$ of pairwise disjoint non-empty subsets of V whose union is V is called a *partition* of V. $\mathscr{P}$ is said to *separate* S or to be *S-separating* if each member of $\mathscr{P}$ contains exactly one element of S. The *value* $e_G(\mathscr{P})$ of $\mathscr{P}$ is the number of edges connecting distinct parts. Clearly, $e_G(\mathscr{P})=\sum_{X\in\mathscr{P}} d(X)/2$ where $d(X)$ denotes the number of edges leaving X. The set of edges connecting distinct members of an S-separating partititon $\mathscr{P}$ is called a *multiway cut* (separating S). We are interested in the minimum cardinality π_S of a multiway cut, that is, in an S-separating partition of minimum value.

Minimum multiway cuts have been subject of study before, see, e.g. [1–3]. The minimum multiway cut problem was shown to be NP-complete even for some restricted

* Corresponding author. E-mail: laszlo@math.sc.edu.

[1] Research supported by the Hungarian National Foundation for Scientific Research Grant, OTKA T016358.

[2] Research supported by the Hungarian National Foundation for Scientific Research Grant, OTKA T17580, Hungary.

[3] Research supported by the Hungarian National Foundation for Scientific Research Grant, OTKA T016358 and by NSF DMS 9701211.

0166-218X/98/$19.00
PII S0166-218X(98)00049-3

versions [4]. Paper [5] introduced a new lower bound for the minimum cardinality of a multiway cut and proved a min–max theorem in the special case when the nodes of S cover all circuits. For this special class, even the weighted multiway cut problem has been solved in [6].

In the first section we introduce a new lower bound $\vec{\nu}_S$ for the minimum multiway cut. This value is the same as the lower bound used in [5] but its definition is more transparent. We compare $\vec{\nu}_S$ with two known lower bounds and prove that, among these three, $\vec{\nu}_S$ is always the best (that is, the largest). Section 2 includes the main contribution of the paper: it is a relatively simple algorithmic proof for a simlified version of a min–max theorem of [5]. This result is about undirected graphs and the basic idea behind the present simplification is that we introduce orientations of the underlying undirected graph.

We need the following notions and notation. A *leaf* of a tree is a node of degree one. A *star* is a tree whose all but possibly one nodes are leaves. We call a directed tree $\vec{T}$ an *arborescence* if every node is reachable by a directed path from a special node, called the root of $\vec{T}$.

Given a hypergraph $\mathscr{L}$, the *degree* of a node u is the number of members of $\mathscr{L}$ containing u. For a subset Z of nodes of a graph $G=(V,E)$, the set of edges connecting Z and $V-Z$ is called a *cut*. It is denoted by $[Z,V-Z]$ and its cardinality by $d(Z)=d_G(Z)$. For a digraph $\vec{G}$ let $\varrho(Z)=\varrho_{\vec{G}}(Z)$ denote the number of edges entering Z. For two disjoint subsets A,B of V, let $\lambda(A,B;\vec{G})$ denote the maximum number of edge-disjont (directed) paths with starting node in A and end node in B. By Menger's theorem $\lambda(A,B;\vec{G})=\min(\varrho(X): B\subseteq X\subseteq V-A)$. For $s\in S$ let $\lambda(S-s,s;G)$ denote the maximum number of edge-disjoint paths from $S-s$ to s. If $\vec{G}$ is a directed graph we use the notation $\lambda(S-s,s;\vec{G})$ for the maximum number of edge-disjoint directed paths from $S-s$ to s. Note that via the Max-flow Min-cut algorithm both $\lambda(S-s,s;G)$ and $\lambda(S-s,s;\vec{G})$ are computable in polynomial time.

1. Lower bounds

First, we try to find some lower bounds for π_S. Let $\tau_S^*:=\sum_{s\in S}\lambda(S-s,s;G)/2$. The quantity τ_S^* was introduced by Lovász [7]. He proved that τ_S^* is equal to the maximum value of a fractional packing of S-paths and also to the minimum value of a fractional edge-covering of S-paths, where an S-path is a path connecting two distinct elements of S. For τ_S^* one has $\tau_S^*=\sum_{s\in S}\lambda(S-s,s;G)/2\leqslant\sum_{s\in S}d(V_s)/2=e_G(\mathscr{P})$ for any S-separating partition $\mathscr{P}=\{V_s: s\in S\}$, from which $\tau_S^*\leqslant\pi_S$ follows. Therefore, τ_S^* is a polynomially computable lower bound for π_S. It is not a very good one though as is shown by a star with k leaves where S consists of the k leaves. For such a star $\tau_S^*=k/2$ and $\pi_S=k-1$.

In order to obtain better bounds we introduce two other parameters. By the *value* $\mathrm{val}(T)$ of a sub-tree T of G we mean the number of its leaves belonging to S minus one. In particular, the value of a path connecting two elements of S is 1.

Let ν_S^{tree} denote the maximum sum of values of edge-disjoint trees of G. Let $\vec{\nu}_S := \max(\sum_{s\in S}\lambda(S-s,s;\vec{G}))$ where the maximum is taken over all orientations $\vec{G}$ of G.

Theorem 1.1. $\tau_S^* \leqslant \nu_S^{\text{tree}} \leqslant \vec{\nu}_S \leqslant \pi_S$.

Proof. To see the last inequality, suppose that $\vec{G}$ is an orientation of G for which $\vec{\nu}_S = \sum_{s\in S}\lambda(S-s,s;\vec{G})$ and that $\mathscr{P} := \{V_s\colon s\in S\}$ an S-separating partition for which $e_G(\mathscr{P}) = \pi_S$. Then

$$\vec{\nu}_S = \sum_{s\in S}\lambda(S-s,s;\vec{G}) \leqslant \sum_{s\in S}\varrho(V_s) = \sum_{s\in S} d(V_s)/2 = e_G(\mathscr{P}) = \pi_S,$$

as required.

The middle inequality is also straightforward. Indeed, let $T_1, T_2, \ldots, T_k$ be the members of an optimal packing of trees. Orient the edges of each T_i as follows. Choose arbitrarily a leaf of T_i in S and orient each edge of T_i so as to obtain an arborescence with this root. The edges not in any T_i may be oriented arbitrarily. In such an orientation $\vec{G}$ of G the value $\lambda(S-s,s;\vec{G})$ is at least as large as the number of trees containing s whose chosen root is different from s. Therefore, the sum $\sum_{s\in S}\lambda(S-s,s;\vec{G})$ is at least the sum of the values of the trees. We obtain, that $\vec{\nu}_S \geqslant \sum_{s\in S}\lambda(S-s,s;\vec{G}) \geqslant \nu_S^{\text{tree}}$.

Finally, we prove the first inequality

$$\tau_S^*(G) \leqslant \nu_S^{\text{tree}}(G). \tag{1.1}$$

By induction, we assume that

(*) inequality (1.1) holds for any graph $G' = (V', E')$ for which $|V'|+|E'| < |V|+|E|$.

We may assume that the deletion of any edge e decreases τ_S^*. Indeed, if the deletion of e leaves τ_S^* unchanged, then by (*) we have $\tau_S^*(G) = \tau_S^*(G-e) \leqslant \nu_S^{\text{tree}}(G-e) \leqslant \nu_S^{\text{tree}}(G)$, as required. We also may assume that there is no edge e connecting two elements of S. Indeed, leaving out such an edge decreases both τ_S^* and ν_S^{tree} by one and hence (*) implies again (1.1).

Case 1: There is a set Z of nodes for which $|Z| \geqslant 2$, $Z \cap S = \{s\}$ for some $s \in S$ and $\lambda(S-s,s;G) = d_G(Z)$.

Contract Z into one node denoted by s_Z. In the contracted graph G' let $S' := S - s + s_Z$. Using (*) and the fact that contraction does not decrease any value $\lambda(S-x,x;G)(x\in S)$, we have $\nu_S^{\text{tree}}(G') \geqslant \tau_{S'}^*(G') \geqslant \tau_{S'}^*(G)$. Therefore, there is a family $\mathscr{T}'$ of edge-disjoint trees in G' so that $\sum(\text{val}(T)\colon T\in\mathscr{T}') \geqslant \tau_S^*(G)$.

We assume that $|\mathscr{T}'|$ is as large as possible. In this case we claim that each terminal node $x \in S'$ belonging to a tree $T \in \mathscr{T}'$ is a leaf of T. For otherwise we could split T at x into $d_T(x)$ subtrees. Then the total value of the new family of trees is unchanged, contradicting the maximality of $|\mathscr{T}'|$, and the claim follows.

Since $\lambda(S-s,s;G) = d_G(Z)$, there is a family of $d_G(Z)$ edge-disjoint paths in G connecting s and $S-s$. For an edge e in the cut $[Z, V-Z]$ let P_e denote the path in this family containing e and let P_e' be the subpath of P_e whose first node is s and last edge is e. If a tree $T' \in \mathscr{T}'$ uses an edge e' of G' corresponding to an edge

$e \in [Z, V-Z]$ of G, then $T := T' - e' + P'(e)$ is a tree of G for which $\mathrm{val}(T) = \mathrm{val}(T')$. In $\mathcal{T}'$ replace each such T' by T.

Every tree in $\mathcal{T}'$ not containing s_Z corresponds to a tree T of G (disjoint from Z) whose value is the same. Therefore, we have obtained a family $\mathcal{T}$ of edge-disjoint trees of G for which $\nu_S^{\mathrm{tree}}(G) \geqslant \sum(\mathrm{val}(T)\colon T \in \mathcal{T}) = \sum(\mathrm{val}(T')\colon T' \in \mathcal{T}') \geqslant \tau_S^*(G') \geqslant \tau_S^*(G)$, as required.

Case 2:

$$\lambda(S-s, s; G) = d_G(s) < d_G(Z) \tag{1.2}$$

holds whenever $s \in S$, $Z \cap S = \{s\}$ and $|Z| \geqslant 2$.

By Menger's theorem, the deletion of an edge e decreases τ_S^* if and only if e belongs to a (minimum) cut $[Z, V-Z]$ for which $Z \cap S = \{s\}$ and $d_G(s) = \lambda(S-s, s; G)$ for some $s \in S$. Therefore, every edge e of G has exactly one end-node in S, that is, G is bipartite.

For each $s \in S$, $v \in V - S$ let $c(sv)$ denote the number of parallel edges between s and v. For $v \in V - s$ let $\alpha(v) := \max(c(sv)\colon s \in S)$. We claim that

$$\alpha(v) < d_G(v)/2, \tag{1.3}$$

for otherwise $d_G(\{s, v\}) \leqslant d_G(s)$, contradicting (1.2). By (1.3) the set of edges incident to v can be partitioned into $\alpha(v)$ stars so that each contains at least two edges. The value of one such star is one less than the number of its edges and hence the total value of the $\alpha(v)$ trees is $d_G(v) - \alpha(v)$. Applying this way of partitioning to each $v \in V - S$, we obtain a family of trees whose total value is $\sum([d_G(v) - \alpha(v)]\colon v \in V-S) > \sum(d_G(v)/2\colon v \in V-S) = |E|/2 = \sum(d_G(s)/2\colon s \in S) = \sum(\lambda(S-s, s; G)/2\colon s \in S) = \tau_S^*(G)$, from which (1.1) follows.

In Theorem 1.1 strict inequality may occur at each place. That was shown already for the first inequality. In the graph in Fig. 1 $\tau_S^* = 3 = \nu_S^{\mathrm{tree}}$ and $\vec{\nu}_S = 4 = \pi_S$, that is the second inequality is strict.

In the first graph in Fig. 2 $\tau_S^* = 6$, $\nu_S^{\mathrm{tree}} = 7 = \vec{\nu}_S$, $\pi_S = 8$. (A tree-packing of total value 7 is shown in the second of Fig. 2. The fact that $\vec{\nu}_S < 8$ can be shown by case checking.)

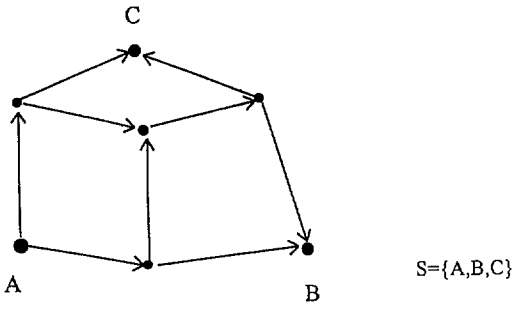


Fig. 1.

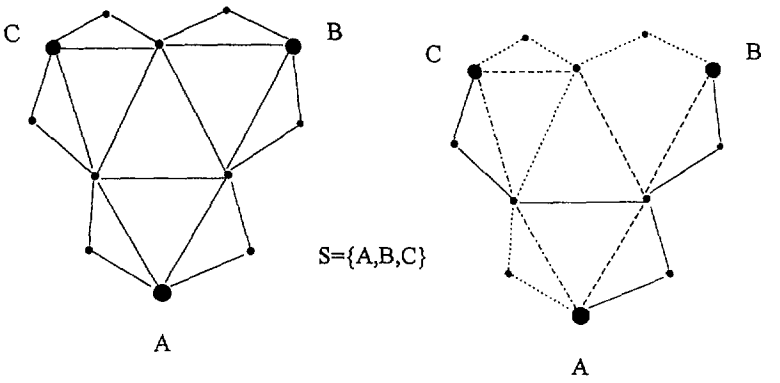


Fig. 2.

2. Min–max theorem and algorithm

The example of Fig. 2 leaves little room to find classes of graphs for which $\vec{\nu}_S = \pi_S$. In what follows, we prove that those graphs for which $G - S$ induces a tree form such a class. (The apparently more general case when $G - S$ induces a forest is easily seen to be equivalent to the tree case.) The theorem below is equivalent to the min–max theorem of [5], but formulated in simpler terms relying on the notion of orientations. This idea gave rise to a proof significantly simpler than the original one. (Actually, the result below extends to the case when $G - S$ may induce only two-element circuits. This is equivalent to the weighted multiway cut problem in trees and was solved in [6]. We hope, though the details have not yet been worked out, that the present orientation method can be extended to the weighted case, as well.)

If one is interested only in computing a minimum multiway cut in the special case when $G - S$ induces a tree, then a simple greedy type algorithm is available in [5] whose proof of correctness is also very simple and does not need any kind of duality theory. The first part of the present algorithm is nothing but a reformulation of the greedy algorithm from [5]. The main novelty here lies in the second part of the algorithm where an optimal orientation is computed yielding a relatively simple proof of Theorem 2.1.

Theorem 2.1. *Let $G=(V,E)$ be an undirected graph with a terminal set S for which $G - S$ induces a tree. Then $\vec{\nu}_S = \pi_S$, that is, the minimum cardinality of a multiway cut separating S is equal to the maximum of $\sum_{s\in S} \lambda(S-s,s;\vec{G})$ over all orientations $\vec{G}$ of G.*

Proof. We have seen that $\vec{\nu}_S \leqslant \pi_S$. In order to prove the equality, we are going to construct a partition $\mathscr{P}$ separating S and an orientation $\vec{G}$ of G so that

$$\varrho_{\vec{G}}(X) = \lambda(S-s,s;\vec{G}) \quad \text{whenever } s \in X \in \mathscr{P}. \tag{2.1}$$

Before proceeding to the proof, let us mention that there is another interpretation of the problem. Consider each element of S as a colour. Then the minimum multiway cut problem is equivalent to colouring the nodes with the available $|S|$ colours so as to minimize the number of bi-chromatic edges. For a given colouration we say that an edge uv is *bi-chromatic* if its two ends have different colours. The other edges are called *mono-chromatic*.

We may assume that S is a stable set. We also may assume, without loss of generality, that for every edge sv with $s \in S$ the degree of v is 2. If this is not the case, then subdivide the edge vs by a new node. Clearly, the theorem holds for the new graph if and only if it holds for the original.

Let $T=(U,F)$ denote the tree induced by $G-S$. By the assumptions we made, only the leaves of T have neighbours in S. We may furthermore assume that every leaf v actually has one neighbour in S for otherwise we may delete v without changing the problem.

Let us choose an arbitrary non-leaf node r of T and call it a *root*. The *height* $h(u)$ of a node u of T is the length of the unique path from r to u. For an edge $e=uv$ with $h(u)=h(v)-1$ we say that node v and edge e are *above* u and that u and e are *under* v. That is, the nodes above u are exactly those neighbours of u whose height is one bigger than that of u. Furthermore, every node u but the root has exactly one node under u. There is no node under the root and above a leaf. (In the literature a node above u is called a child of u and a node under u is called a parent of u.)

With the help of a depth first search (say), determine an ordering of the elements of U, described by a one-to-one mapping $f: U \to \{1,2,\ldots,|U|\}$, in such a way that $f(r)=1$ and $f(u)<f(v)$ whenever uv is an edge of T with $h(u)=h(v)-1$.

The algorithm consists of two parts. In the first one we determine a partition $\mathscr{P}$ of V separating S while the second part serves for computing the orientation.

Part 1: Computing partition $\mathscr{P}$. The partition $\mathscr{P}$ will be given by a function $\sigma: V \to S$ such that $\sigma(s):=s$ for $s \in S$. (In other words $\sigma(u)$ will be the colour of u.) The σ-values of the tree are computed in two phases.

In the first phase a subset $L(u)$ of S will be assigned to every node u of T, as follows. According to the ordering f of U, consider the elements u of U in a reverse order (that is, root r is considered last). If u is a leaf whose unique neighbour in S is s, then let $L(u):=\{s\}$. Suppose that u is a node for which $L(v)$ has been computed for all nodes v above u. Let $L(u)$ consist of the nodes of maximum degree of the hypergraph $\{L(v)\colon v \text{ is above } u\}$. The first phase terminates when $L(r)$ (and hence every other $L(v)$) has been computed.

Intuitively, we think of $L(u)$ as the set of candidate colours from which the final colour of u will be chosen during the second phase of Part 1. The sets $L(u)$ are determined in a downward manner (toward the root) and $L(u)$ consists of those colours which appear most often in the (already determined) candidate colour-sets of nodes above u.

In the second phase we work upward, that is we consider the elements v of U in the (forward) ordering given by f. Start at the root r and define $\sigma(r)$ to be an arbitrary member of $L(r)$. In the general step, when v is considered let uv denote the unique edge of T under v. By the choice of the ordering, u precedes v and hence $\sigma(u)$ has already been determined.

(a) If $\sigma(u) \in L(v)$, then let $\sigma(v) := \sigma(u)$,
(b) if $\sigma(u) \notin L(v)$, then let $\sigma(v)$ be an arbitrary member of $L(v)$.

Intuitively, this means that the colour of root r is an arbitrary member of the candidate colour-set $L(r)$. Furthermore, if the colour $\sigma(u)$ of a node of tree T has already been determined and $e = uv$ is an edge of T above u, then the colour $\sigma(v)$ of v is always chosen from the candidate colour-set $L(v)$ of v so as to make e mono-chromatic whenever this is possible. (That is, e becomes bi-chromatic if the final colour $\sigma(u)$ of u is not in the set $L(v)$ of candidate colours of v.) This way, we have determined a colouration σ of the nodes of G, or, equivalently, a partition $\mathscr{P} := \{V_s: s \in S\}$ of V where $V_s := \{u \in V: \sigma(v) = s\}$.

Part 2. Computing the orientation of T. In the second part of the algorithm we define the orientation of the edges of G in such a way that once the orientation of an edge has been determined, it will never be changed later. Let $e = uv$ be an edge of T with $h(u) = h(v) - 1$. The orientation of e will be specified by declaring that e is either an up-edge or a down-edge. e being an *up-edge* means that e is oriented from u to v while e being a *down-edge* means that e is oriented from v to u. We will call a node v distinct from the root an *up-node* if the (unique) edge under v is an up-edge. Node v is called a *down-node* if either $u = r$ or if the edge under u is a down-edge.

Let all bi-chromatic edges be up-edges. To determine the orientation of other edges, we consider the nodes u of T in the order of their height starting with root r and determine the orientation of all the mono-chromatic edges above u. Therefore, when a node u is considered, its status of being an up-node or a down-node has already been determined. Specifically, the orientation of the mono-chromatic edges above u is determined by the following rules.

Rule 1: If u is a down-node, then let every mono-chromatic edge above u be a down-edge.

Rule 2: If u is an up-node, then choose arbitrarily a mono-chromatic edge uz above u (there is one!) and, apart from uz, let all mono-chromatic edges above u be down-edge. We will call uz a *special* edge.

Rule 3: If e is an edge connecting a leaf u of T and a node s in S, then orient e so that the in-degree (and the out-degree) of u be 1.

In other words, every bi-chromatic edge is an up-edge and every mono-chromatic edge above u, with one exception in case u is a down-node, is a down-edge.

Let $\vec{G}$ denote the resulting directed graph. Henceforth, our main concern is to prove that the partition $\mathscr{P}$ and the orientation $\vec{G}$ satisfy (2.1). To this end let us consider an element $s \in S$ (that is, one of the colours) and the set $F_s := \{u_1v_1, \ldots, u_kv_k\}$ of

bi-chromatic edges of $\vec{G}$ for which $\sigma(v_i)=s$. That is, F_s is the set of edges of $\vec{G}$ entering the part V_s of $\mathscr{P}$ containing s.

We are going to find k edge-disjoint paths from $S-s$ to s. The existence of such paths directly implies (2.1). It follows from Rule 2 that for any up-node v_i $(i=1,\ldots,k)$ there is a unique path P_i' in $\vec{G}$ from v_i to s consisting of special edges. Since special edges are mono-chromatic these paths are inside V_s. They are edge-disjoint (and actually node-disjoint, except at s) since no two special edges enter the same node.

Therefore, all what we have to show is that there are k edge-disjoint paths P_i'' from $S-s$ to v_i $(i=1,\ldots,k)$. By glueing together paths P_i' and P_i'' we will obtain a path P_i from a node of $S-s$ to s that uses edge u_iv_i.

Let $\vec{G}_s=(V,E_s)$ be a subgraph of $\vec{G}$ where E_s consists of three types of edges. Recall that if (a directed edge) yv is a down-edge or vy is an up-edge, then y is above v.

Type A: A down-edge yv belongs to E_s if $s\notin L(y)$.

Type B: An up-edge vy belongs to E_s if $s\in L(y)$ and $\sigma(v)\neq s$.

Type C: An edge tu of $\vec{G}$ belongs to E_s if $t\in S-s$.

Note that a down-edge $yv\in E_s$ is mono-chromatic and $\sigma(v)=\sigma(y)\neq s$. Hence, $\sigma(v)\in L(y)$. For a non-special up-edge vy, $\sigma(v)\notin L(y)$.

Let $\varrho(u)$ (respectively, $\delta(u)$) denote the number of edges in E_s entering (leaving) u.

Lemma 2.2. *$\varrho(u)\geqslant\delta(u)$ for every node u of T.*

Proof. By Rule 3, the lemma holds for leaves so suppose that u is not a leaf. If $\sigma(u)=s$, then, by Rules 1 and 2, $\delta(u)=0\leqslant\varrho(u)$. Therefore, we will assume that $\sigma(u)\neq s$. Let $A:=\{y\in U:\ y$ is above $u, s\in L(y), \sigma(u)\notin L(y)\}$ and $B:=\{y\in U:\ y$ is above $u, s\notin L(y)$, $\sigma(u)\in L(y)\}$. Let $\alpha:=|A|$, $\beta:=|B|$.

Since $\sigma(u)\in L(u)$, the definition of $L(u)$ implies that

$$\beta\geqslant\alpha \quad \text{and if } \beta=\alpha, \quad \text{then } s\in L(u). \tag{2.2}$$

Let x denote the node under u in case $u\neq r$.

Case 1: u is a down-node. Then the edges of E_s above u that leave u are precisely the edges from u to A. Hence, $\delta(u)\leqslant\alpha+1$ and $\delta(u)=\alpha$ if u is the root. Each edge under an element of B is a down-edge and belongs to E_s from which $\varrho(u)\geqslant\beta$. If $u=r$, then $\varrho(u)\geqslant\beta=\alpha=\delta(u)$, as required. So suppose that $u\neq r$. If $\beta\geqslant\alpha+1$, then $\varrho(u)\geqslant\delta(u)$. If $\alpha=\beta$, then (2.2) implies that $ux\notin E_s$ and hence $\varrho(u)\geqslant\beta=\alpha=\delta(u)$.

Case 2: u is an up-node. If $\alpha=\beta$, then $s\in L(u)$ by (2.2). Now $\sigma(x)\neq s$, since $\sigma(x)=s$ would imply $\sigma(u)=s$ which is not the case. Therefore, $xu\in E_s$.

Let uz denote the special edge above u. Now, $\sigma(u)=\sigma(z)\in L(z)$ and hence $z\notin A$. We distinguish two cases.

If $z\in B$, then $s\notin L(z)$ and hence $uz\notin E_s$. Therefore, the edges in E_s leaving u are the edges from u to A, that is, $\delta(u)=\alpha$. For every node $y\in B-z$ the edge under y is oriented toward u. If $\beta\geqslant\alpha+1$, then $\varrho(u)\geqslant\beta-1\geqslant\alpha=\delta(u)$. If $\beta=\alpha$, then $xu\in E_s$ and hence $\varrho(u)\geqslant(\beta-1)+1=\alpha=\delta(u)$.

If $z \notin B$, then $\delta(u) \leqslant \alpha + 1$. If $\beta \geqslant \alpha + 1$, then $\varrho(u) \geqslant \beta \geqslant \alpha + 1 \geqslant \delta(u)$. If $\beta = \alpha$, then $xu \in E_s$ and hence $\varrho(u) \geqslant \beta + 1 = \alpha + 1 \geqslant \delta(u)$. □

Now, we can construct the paths P_i'' $(i = 1, \ldots, k)$ in a greedy way. Starting at $u_1 v_1$ we can go backward in E_s as long as we arrive at a node of S which is distinct from s since the tail of every edge in E_s has got a colour distinct from s. That is, we have constructed a directed path P_1'' that starts at an element of $S - s$ and its last edge is $u_1 v_1$. After leaving out the edges of this path the property of Lemma 2.2 continues to hold, so we can repeat the construction to obtain the required edge-disjoint paths $P_1'', P_2'', \ldots, P_k''$.

References

[1] D. Bertsimas, C. Teo, R. Vohra, Nonlinear formulations and improved randomized algorithms for multicut problems, in: E. Balas, J. Clausen (Eds.), Integer Programming and Combinatorial Optimization, 4th International IPCO Conf., Copenhagen, Denmark, May 1995, Proceedings, Lecture Notes in Computer Science, vol. 920, Springer, Berlin, pp. 29–39.

[2] S. Chopra, M. Rao, On the multiway cut polyhedron, Networks 21 (1991) 51–89.

[3] W.H. Cunningham, The optimal multiterminal cut problem, in: DIMACS Series Disc. Math. 5 (1991) 105–120.

[4] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, M. Yannakakis, The complexity of multiway cuts, 24th ACM STOC, 1992, pp. 241–251, see also E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, M. Yannakakis, The complexity of multiterminal cuts, SIAM J. Comput. 23 (1994) (4) 864–894.

[5] P.L. Erdős, L.A. Székely, Evolutionary trees: an integer multicommodity max-flow–min-cut theorem, Adv. Appl. Math. 13 (1992) 375–389.

[6] P.L. Erdős, L.A. Székely, On weighted multiway cuts in trees, Math. Programming 65 (1994) 93–105.

[7] L. Lovász, On some connectivity properties of Eulerian graphs, Acta Math. Acad. Sci. Hungar. 28 (1976) 129–138.

Computers and Artificial Intelligence, Vol. 16, 1997, No. 2, 217—227

LOCAL QUARTET SPLITS OF A BINARY TREE INFER ALL QUARTET SPLITS VIA ONE DYADIC INFERENCE RULE[1]

Péter L. ERDŐS

Mathematical Institute of the Hungarian Academy of Sciences
P.O.Box 127, Budapest, Hungary-1364
E-mail: elp@math-inst.hu

Michael A. STEEL

Biomathematics Research Centre, University of Canterbury
Christchurch, New Zealand
E-mail: m.steel@math.canterbury.ac.nz

László A. SZÉKELY

Department of Mathematics, University of South Carolina
Columbia SC 29208, USA.
E-mail: laszlo@math.sc.edu
Department of Computer Science, Eötvös University
Budapest, Hungary-1088.
E-mail: szekely@cs.elte.hu

Tandy J. WARNOW

Department of Computer and Information Science
University of Pennsylvania
Philadelphia PA 19104-6389, USA
E-mail: tandy@central.cis.upenn.edu

Abstract. A significant problem in phylogeny is to reconstruct a semilabelled binary tree from few valid quartet splits of it. It is well-known that every semilabelled binary tree is determined by its set of all valid quartet splits. Here we strengthen this result by showing that its local (i.e. small diameter) quartet splits infer by a dyadic inference rule all valid quartet splits, and hence determine the tree. The results of the paper also present a polynomial time algorithm to recover the tree.

Keywords: semilabelled binary trees, subtrees, phylogeny, quartets.

1 INTRODUCTION

We first provide a summary of notations used throughout this paper. The set $[n]$ denotes $\{1, 2, \ldots, n\}$ and for any set S, $\binom{S}{k}$ denotes the collection of subsets of S of size k.

A *semilabelled binary tree* T is a tree whose *leaves* (vertices of degree 1) are labelled by the number $1, 2, \ldots, n$, and whose remaining internal vertices are unlabelled and of degree three. Let $B(S)$ denote the set of semilabelled binary trees on leaf set S, and let $B(n) = B([n])$. For $T \in B(n)$ and $S \subseteq [n]$, there is a unique minimal subtree of T which contains all the elements of S. We call this tree the *subtree of* T *induced by* S, and denote it by $T_{|S}$. We obtain the *binary subtree of* T *induced by* S, denoted by $T^*_{|S}$, if we substitute edges for all maximal paths of $T_{|S}$ in which every internal vertex has degree two. Thus, $T^*_{|S} \in B(S)$. If $|S| = k$, then we refer to $T^*_{|S}$ as a *binary* k*-subtree.*

Given a semilabelled binary tree T with leaf set S, deleting an edge e of T disconnects T into two components, and thereby induces a bipartition of S consisting of the leaves of the two components. This bipartition is called a *split* of T induced by the edge e; the split is called *non-trivial* if both components contain at least 2 leaves. Buneman [3] showed that each semilabelled binary tree T is uniquely defined by its non-trivial splits.

[1] **Acknowledgment**. This research started when the authors enjoyed the hospitality of DIMACS during the Special Year for Mathematical Support to Molecular Biology. The second author gratefully acknowledges the New Zealand Ministry of Research, Science and Technology (MORST) for support to visit Budapest under ISAC Programme grant 94/22. Research of the first and third authors was supported in part by the Hungarian National Science Fund contract T 016 358 and by the European Communities (Cooperation in Science and Technology with Central and Eastern European Countries) contract ERBCIPACT 930 113.The fourth author was supported in part by a Young Investigator Award from the National Science Foundation, CCR-9457800, and by grant SBR-9512092 from the Linguistics program at NSF, and by generous financial support from Paul Angello.

For a semilabelled binary tree $T \in B(n)$, and for a quartet of leaves, $q = \{a, b, c, d\} \in \binom{[n]}{4}$, we say that $t_q = ab|cd$ is a *valid quartet split* of T, if $ab|cd$ is a split of $T^*_{|q}$. It is easy to see that

$$\text{if } ab|cd \text{ is a valid quartet split of } T, \text{ then so are } ba|cd \text{ and } cd|ab, \tag{1}$$

and we understand these three splits as identical. If (1) holds, then $ac|bd$ and $ad|bc$ are not valid quartet splits of T, and we say that any of them *contradicts* (1).

2 TREE RECONSTRUCTION FROM AN INCOMPLETE SET OF VALID QUARTET SPLITS

Let $Q(T) = \left\{t_q : q \in \binom{[n]}{4}\right\}$ denote the set of valid quartet splits of T. It is a classical result that $Q(T)$ determines T (Colonius and Schulze [4], also Bandelt and Dress [1]); indeed for each $i \in [n]$, $\{t_q : i \in q\}$ determines T, and T can be computed in polynomial time. For example, a simple algorithm for reconstructing T from $Q(T)$ is simply to build up T recursively from the tree with leaf set 1,2,3 by attaching (in any order) the remaining elements from $[n]$ as new leaves to the tree so far constructed. In this way, one uses $Q(T)$ to determine the unique edge of each partial tree to which the new leaf must be attached by bisecting the edge and making the recently created vertex adjacent to the new leaf.

An extension of Colonius and Schultze's result [4] is that for any $T \in B(n)$, a carefully chosen subset of $Q(T)$ of cardinality $n - 3$ determines T (Steel [9]). Another extension is that an unknown semilabelled binary tree T with n leaves can be constructed by asking at most $O(n \log n)$ queries of the form: "what is t_q?" for a choice of q that depends on the answers to the queries so far asked (Pearl and Tarsi [7], Kannan, Lawler, and Warnow [6]).

It would be useful to tell from a set of quartet splits if they are valid quartet splits of any semilabelled binary tree. Unfortunately, this problem is NP-complete (Steel [9]). It also would be useful to know which subsets of $Q(T)$ determine T and which subsets would allow for a polynomial time procedure to reconstruct T. A natural step in this direction is to define *inference*: a set of quartet splits A infers a quartet split t_q if whenever $A \subseteq Q(T)$ for a semilabelled binary tree T, then $t_q \in Q(T)$ as well.

Setting a complete list of inference rules seems hopeless (Bryant and Steel [2]). However, having just some valid quartet splits of T, it is often possible to infer additional valid quartet splits of T, for example (see [1], [2] or [5]):

$$\begin{aligned}&\text{if } ab|cd \text{ and } ac|de \text{ are valid quartet splits of } T,\\ &\text{then so are } ab|ce,\ ab|de, \text{ and } bc|de;\end{aligned} \tag{2}$$

if $ab|cd$ and $ab|ce$ are valid quartet splits of T, then so is $ab|de$; (3)

if $ab|cd$, $ab|ef$ and $ce|df$ are valid quartet splits of T, then so is $ab|df$. (4)

In (2) and (3) we infer a valid quartet split from *two* other quartet splits. These rules are called *second order* or *dyadic* rules. In (4) we see a *third order* rule. These rules are due to Dekker [5]. A set of quartet splits A *dyadically infers* a quartet split t, if t can be derived from A by repeated applications of rules (1), (2) and (3).

It is worth mentioning that for every integer r there are inference rules of order r that cannot be inferred by repeated application of lower-order inference rules. (See Dekker [5] and Bryant and Steel [2].)

We say that a set of quartet splits A *semidyadically infers* a quartet split t, if t can be derived from A by repeated applications of rules (1) and (2). Quartet splits (semi)dyadically inferred by a set of quartet splits can be computed in polynomial time, and quartet splits (semi)dyadically inferred by a set of valid quartet splits of a tree are valid. We denote by $cl_2(A)$ the set of all quartet splits semidyadically inferred by the set A of quartet splits. We say that a set of quartet splits A *(semi)dyadically determine* T if they (semi)dyadically infer *all* valid quartet splits of T, i.e. $Q(T)$; in other words, Q fully determines the tree T.

3 TREE RECONSTRUCTION FROM LOCAL QUARTETS

For a semilabelled binary tree $T \in B(n)$, and a quartet of leaves, $q \in \binom{[n]}{4}$, let $L_T(q, e)$ denote the *length* (the number of edges) of the path P_e of $T_{|q}$ which turned into the edge e of $T^*_{|q}$. We will abuse the notation somewhat and let $L_T(q)$ denote the length of the longest path of $T_{|q}$ which is turned into an edge of $T^*_{|q}$, i.e. $L_T(q) = \max_e L_T(q, e)$. In [10] Steel *et al.* proved the following extension of the classical result of Colonius and Shultze:

Theorem 1. For a semilabelled binary tree T on $[n]$ ($n \geq 4$), let

$$D(T) = \{q \in \binom{[n]}{4} : L_T(q) \leq 18 \log n\}.$$

Then $S(T) = \{t_q$ valid quartet split of T : $q \in D(T)\}$ semidyadically determines T. In particular, T can be reconstructed from $S(T)$ in polynomial time.

The interesting point in this proposition is that the local quartets fully determine the underlying binary tree. Based on this fact we built a reconstruction method for Cavender–Farris trees (see [10]). Our main goal is to strengthen Theorem 1. We need some more definitions.

The *depth* of an edge e in a semilabelled binary tree T is the number of edges on the path from e to the nearest leaf. The *depth* of T, $d(T)$, is the maximum depth

of any edge e in T. For example, the depth of a complete semilabelled binary tree on n leaves is $\lceil \log_2 n \rceil$. By contrast, a *caterpillar* on n leaves (the tree defined by a path $p = v_1, v_2, \ldots, v_{n-2}$ in which v_1 and v_{n-2} each has two adjacent leaves and the neighbor of each remaining nodes on p is a leaf) has depth 1.

A *cherry* in a binary tree is a pair of leaves sharing a common neighbor, i.e. a pair of leaves at distance two in the tree.

The following theorem is the main result of this paper:

Theorem 2. For a semilabelled binary tree T on $[n]$, let

$$D(T) = \left\{ q \in \binom{[n]}{4} : L_T(q) \leq 2d(T) + 1 \text{ and } L_T(q, e) = 1, \right.$$
$$\left. \text{where } e \text{ is the internal edge of } T^*_{|q} \right\}.$$

Then $p(T) := \{T^*_{|q} : q \in D(T)\}$ semidyadically determines T. In particular, T can be reconstructed from $p(T)$ in polynomial time.

Proof. We use induction on n. The result holds for $n = 4$, so we suppose $n > 4$. We distinguish two cases:

(a) Every leaf of T is in a cherry, i.e. the leaves of T can be matched $(l_1, l_2), \ldots, (l_{n-1}, l_n)$, such that every pair (l_{2i-1}, l_{2i}) forms a cherry.

(b) There is a leaf l not covered by any cherry, i.e. l is separated from any other leaf by at least three edges.

In Case (a), let λ_i be the common neighbour of the leaves l_{2i-1} and l_{2i}. The deletion of all leaves of T results in a subtree T', whose leaves are just the λ_i's. Note that if E denotes the set of the T-leaves, $E = \{l_{2i} : i = 1, \ldots, n/2\}$, then T' is isomorphic to $T^*_{|E}$. It is clear that $d(T') = d(T) - 1$.

During the proof we assign to quartet splits of T' certain quartet splits of T, and call this operation *extension*. (The point of definition is to extend a valid quartet split into valid quartet splits.)

For a quartet of T'-leaves, $q' \in Q(T')$, where $t_{q'} = \lambda_a\lambda_b|\lambda_c\lambda_d$, we define the *standard general extension* of $t_{q'}$ by the quartet split $t_q = l_{2a}l_{2b}|l_{2c}l_{2d} \in Q(T)$. Now for any quartet of T'-leaves $q' \in D(T')$, we have

$$L_T(q) \leq L_{T'}(q') + 1 \leq [2(d-1) + 1] + 1 < 2d + 1,$$

and, if e is the internal edge of $T^*_{|q}$, then $L_T(q, e) = L_{T'}(q', e) = 1$. Thus the standard general extension t_q of the valid quartet split $t_{q'} \in p(T')$ belongs to $p(T)$. We define the *non-standard general extensions* of the valid quartet split $t_{q'}$ similarly, but we allow the substitution of one or more l_{2j} with l_{2j-1}. It is clear that every

non-standard general extension belongs to $p(T)$ as well. Therefore if $p'(T)$ is the set of all general extensions of $t_{q'} \in p(T')$, then $p'(T)$ is a subset of $p(T)$.

For each leaf λ_j of T', let X_j, Y_j denote the leaf sets of the two other rooted subtrees of T' incident with the unique neighbour of l_j in T', v_j. Since $p(T')$ determines T', there is a quartet q'_j in $D(T')$ containing λ_j, λ_{x_j} and λ_{y_j} where $\lambda_{x_j} \in X_j$ and $\lambda_{y_j} \in Y_j$. We define the *standard special extension* of $t_{q'_j}$ by $t_{q_j} = l_{2j-1}l_{2j}|l_{2x_j}l_{2y_j}$. It is easy to see that

$$L_T(q_j) \leq L_{T'}(q'_j) + 1 \leq [2(d-1)+1] + 1 < 2d + 1.$$

In addition, if e denotes the internal edge of $T^*_{|q_j}$ then $L_T(q_j, e) = 1$. Thus $t_{q_j} \in p(T)$ holds. We define the *non-standard special extensions* of the previous valid quartet split $t_{q'_j}$ similarly, but l_{2x_j} may be substituted by l_{2x_j-1} and/or l_{2y_j} may be substituted by l_{2y_j-1}. All the non-standard special extensions belong to $p(T)$ as well. Let $p^*(T)$ denote the set of all special extensions of $t_{q'_j}$ for every $j = 1, 2, \ldots, n/2$. Then $p^*(T) \subseteq p(T)$. Therefore

$$cl_2\left((p'(T) \cup p^*(T)\right) \subseteq cl_2(p(T)). \tag{5}$$

To finish the proof in Case (a), we now show that the left-hand side of (5) equals $Q(T)$, so that $cl_2(p(T)) = Q(T)$, as claimed. For this purpose, let $t_q = l_a l_b | l_c l_d$ denote an arbitrary valid quartet split in T. Let $\lambda_a, \lambda_b, \lambda_c$ and λ_d be the neighbours of these T-leaves, respectively. If these four T'-leaves are pairwise distinct, then $t_{q'} = \lambda_a\lambda_b|\lambda_c\lambda_d$ is a valid split, and since (by hypothesis) $cl_2(p(T')) = Q(T')$, there is a sequence of inferences in T' yielding $t_{q'} \in Q(T')$ from $p(T')$, using rules (1) and (2). Repeating the same sequence of inferences with the general extensions of these quartet splits (and working in $Q(T)$), we infer t_q as well.

If $\lambda_a = \lambda_b = \lambda_j$ (where j is an integer between 1 and $n/2$), then for every $\lambda_c \in X_j$ the valid quartet split $l_{2j}l_{2j-1}|l_c l_{2y_j}$ belongs to the left-hand side of (5). If the neighbour of l_c happens to be λ_{x_j}, then this is true by the definition of the special extension. So we may assume that $\lambda_c \neq \lambda_{x_j}$. By the preceding part of this case analysis, the valid quartet split $l_{2x_j}l_c|l_{2y_j}l_{2j}$ belongs to the left-hand side of (5) (the neighbours of the four leaves are pairwise distinct). Using rule (1) for the special extension t_{q_j}, we infer $l_{2x_j}l_{2y_j}|l_{2j}l_{2j-1}$. The application of the third consequence in rule (2) infers $l_c l_{2y_j}|l_{2j}\lambda_{2j-1}$. Finally, a second application of rule (1) gives the required valid quartet split.

Similarly, the valid quartet split $l_{2j}\lambda_{2j-1}|l_c l_{2x_j}$ (again, $\lambda_c \neq \lambda_{x_j}$) belongs to the left hand side of (5), as it is shown by the application of the same second order inference rule for $l_{2x_j}l_c|l_{2y_j}l_{2j}$ and for the "opposite" of the valid quartet split t_{q_j}.

If we change the role of λ_{x_j} and λ_{y_j}, we obtain analogous inferences. (Namely, we can infer the quartet splits $l_{2j}l_{2j-1}|l_c l_{2y_j}$, where $\lambda_c \neq \lambda_{y_j}$.) Furthermore, since in the use of inference rules λ_{x_j} and λ_{y_j} do not play any special role, changing the

role of l_{2x_j} (or l_{2y_j}) with an arbitrary leaf $l_d \in X_j$ (or $\in Y_j$, respectively), then we obtain analogous inferences. Therefore the only remaining case is $\lambda_c = \lambda_d$. Without loss of generality we may assume that $\lambda_c \in X_j$. Due to the previous argument, we have already inferred $l_{2j}l_{2j-1}|l_{2y_j}l_c$ and $l_{2j}l_{2y_j}|l_cl_d$. The application of the second consequence of the inference rule (2) infers $l_{2j}l_{2j-1}|l_cl_d$, which finishes the proof of Case (a).

In Case (b), we use the following notations: let l denote a leaf not covered by any cherry, let Λ be the neighbour of l, and let Δ and Γ be the other two neighbours of Λ (by the choice of l, these vertices exist and are of degree three). Let the two subtrees attached to Δ and disjoint from Λ be denoted A and B, and assume that the number of leaves in A is at most the number of leaves in B. Similarly, let the two subtrees attached to Γ be C and D, and assume that the leaves in C is at most the number of leaves in D.

Let the semilabelled binary trees T_Γ and T_Δ be defined in the following way: let T_Γ be the semilabelled binary tree generated by A, B, and the leaves l and Γ. The semilabelled binary tree T_Δ is generated by C, D, and the leaves l and Δ.

By induction, $cl_2(p(T_i)) = Q(T_i)$ hold for $i = \Delta, \Gamma$. Let the leaf $c \in C$ be the closest leaf to Γ from C, and let the leaf $a \in A$ be the closest leaf to Δ from A. Let R_c be obtained from $p(T_\Gamma)$ by omitting the quartet splits of the form $\Gamma x|yz$, where $x \neq l$, and substituting valid quartet splits $\Gamma l|yz \in p(T_\Gamma)$ with quartet splits $cl|yz \in Q(T)$. Similarly, let R_a be obtained from $p(T_\Delta)$ by omitting quartet splits $\Delta x|yz$ for $x \neq l$ and substituting quartet splits $\Delta l|yz$ with $al|yz \in Q(T)$. We define the *lift-up* of valid quartet splits from $p(T_\Delta) \cup p(T_\Gamma)$ considering them being quartets from $Q(T)$, substituting Γ by c and Δ by a whenever necessary, i.e. whenever Γ or Δ belong to the quartets. Therefore, the quartets in R_a and R_c are lift-ups from some quartet splits of the subtrees T_Δ and T_Γ, respectively. We now show that $R_a \cup R_c$ is a subset of $p(T)$. For this purpose, let t_{q_a} be the lift-up of the valid quartet split $t_{q'_a} \in p(T_\Delta)$ and similarly, let t_{q_c} be the lift-up of the valid quartet split $t_{q'_c} \in p(T_\Gamma)$.

It is easy to see that $L_T(q_a) = L_{T_\Delta}(q'_a)$ except for some quartet splits of the form $t_{q_a} = la|\gamma\delta$. Similarly, $L_T(q_c) = L_{T_\Gamma}(q'_c)$ except for some quartet splits of the form $t_{q_c} = cl|\alpha\beta$. What remains is to show that $L_T(la|\gamma\delta) \leq 2d(T) + 1$ and $L_T(cl|\alpha\beta) \leq 2d(T) + 1$. Due to symmetry it is sufficient to prove the first claim only. We will use the notation $t_q = cl|\alpha\beta$.

For the pendant edge e of $T^*_{|q}$ incident with either α and β, we have $L_T(q, e) \leq 2d(T) + 1$ since

$$L_T(q, e) = L_{T_\Gamma}(q, e) \leq L_{T_\Gamma}(q) \leq 2d(T_\Gamma) + 1 \leq 2d(T) + 1.$$

For the edges $e = (\Delta, \lambda)$ and $e = (\lambda, l)$ we have $L_T(q, e) = 1$. Thus, it remains to establish that

$$L_T(q, e) \leq 2d(T) + 1 \tag{6}$$

for the edge e of T_q^* incident with c. If $|C| = 1$, then we have nothing to prove since $L_T(q, e) = 2 \leq 2d(T) + 1$.

Now for $|C| > 1$ suppose on contrary to (6) that $L_T(q, e) > 2d(T) + 1$. Then $d_T(\lambda, c) > 2d(T) + 1$ (where $d_T(x, y)$ denotes the *distance* of x and y in the tree T, that is the length of the path from x to y). Since c is the closest leaf in C to λ, all leaves in C are at distance $> 2d(T) + 1$ from λ. Let $e^* = (x, y)$ be an edge of C for which $d_T(\Gamma, x) = d - 1$ and $d_T(\Gamma, y) = d$. By the definition of $d(T)$, the depth of e^* is at most $d(T)$, therefore there must be a leaf l^* of T at distance at most $d(T)$ from e^*. On the one hand l^* cannot belong to C since all leaves of C must be at distance $> d(T) + 1$ from e^* (by the assumption $L_T(q, e) > 2d + 1$). On the other hand $l^* \neq l$ since the distance $d_T(l, x) = d + 1$. Finally, for every leaf $l^* \in D$, the distance $d_T(l^*, x) > d$ because the path from x to l^* uses at least two edges of D since D has at least two leaves. This contradiction proves that $L_T(q, e) \leq 2d(T) + 1$ and therefore $R_c \subseteq p(T)$, and a similar argument shows that $R_a \subseteq p(T)$. Therefore we have

$$cl_2(R_a \cup R_c) \subseteq cl_2(p(T)). \tag{7}$$

To finish the proof in Case (b), we are going to show that the left-hand side of (7) equals $Q(T)$, so that $cl_2(p(T)) = Q(T)$, as claimed. For this purpose, let b denote the B-leaf in T, which is closest to Δ. Similarly, let d denote the closest leaf to Γ from D in T. We note that the distance $d_T(b, \lambda) \leq 2d(T) + 2$ because we can repeat the proof of formula (6) except that A can be of cardinality one. Therefore $d(b, \Delta) \leq 2d(T) + 1$. A similar condition holds for the leaf $d \in D$ which is the closest one to Γ. From know on, the letters a, b, c and d always refer to these fixed leaves.

At first we show that the valid quartet split $cx|yz \in Q(T)$, which is the lift-up of the quartet split $\Gamma x|yz \in p(T_\Gamma)$, belongs to the LHS of (7). Because $\Gamma x|yz \in p(T_\Gamma)$, therefore if x belongs to A, then $y, z \in B$, and they are on different subtrees of the neighbour Φ of Δ. Furthermore, without loss of generality we may assume that b is on the same subtree of Φ as z.)

We know that $d(x, \Delta) \leq 2d(T) + 1$, since $\Gamma x|yz \in p(T_\Gamma)$. We just show that $d(b, \Delta) \leq 2d + 1$. Therefore the valid quartet split $lc|xb$ belongs to R_c. Similarly, $lx|yb \in R_c$ also holds. Applying the first consequence of inference rule (2), we have $lc|xy \in cl_2(R_c)$. Putting this together with $lx|yz \in R_c$ (which follows from the fact that $d(\Gamma, \Delta) = d(l, \Delta)$) and applying again rule (2), consequence 3, we proved that $cx|yz \in cl_2(R_c)$. The symmetric claim $au|vw \in cl_2(R_a)$ holds for the lift-up of $\Delta u|vw \in p(T_\Delta)$. Thus, we have proved:

$$\begin{aligned} &\text{the lift-up version of any element of } p(T_\Delta) \text{ or } p(T_\Gamma) \\ &\text{belongs to } cl_2(R_a) \text{ or } cl_2(R_c). \end{aligned} \tag{8}$$

Let α and β denote two leaves in $A \cup B$. Since $\alpha\beta|l\Gamma \in Q(T_\Gamma)$, therefore, due to (8), there is a "lifted up" inference sequence for $\alpha\beta|lc$ by semidyadic inference rules.

For $\gamma, \delta \in C \cup D$ we have a similar result. Thus, we have proved:

$$\text{for leaves } \gamma, \delta \in C \cup D \; al|\gamma\delta \in cl_2(R_a), \tag{9}$$

$$\text{for leaves } \alpha, \beta \in A \cup B \; \alpha\beta|lc \in cl_2(R_c). \tag{10}$$

From now on, α, β, γ and δ always refer to leaves like above, but they are not fixed leaves.

Assume that $a \neq \alpha$ (if this is not true, then exchange the names α and β). Similarly, we may assume that $c \neq \delta$. Applying the choice $\beta = a$, for property (10), we have $a\alpha|lc \in cl_2(R_c)$. Similarly, for $\gamma = c$ and $\delta = d$ in (9) we have $al|cd \in cl_2(R_a)$. The application of the first consequence of rule (2) gives:

$$a\alpha|ld \in cl_2(R_a \cup R_c). \tag{11}$$

The substitution $\delta = d$ in (9) gives $al|\gamma d \in cl_2(R_a)$. This, together with (11), through the application of the third consequence of (2) gives:

$$\alpha l|\gamma d \in cl_2(R_a \cup R_c). \tag{12}$$

(10) together with (12) (where $\gamma = c$) gives (through rule (2), first consequence)

$$\alpha\beta|ld \in cl_2(R_a \cup R_c). \tag{13}$$

Applying the symmetry rule (1) for (12) and (13) and using again the semidyadical rule (2) with its third consequence and taking again its symmetric form, we have

$$\alpha\beta|l\gamma \in cl_2(R_a \cup R_c). \tag{14}$$

Since γ was not involved in the proof of (14) (except that $\gamma \in R_a \cup R_c$), the following symmetric claim can also be inferred through similar reasoning:

$$\alpha l|\gamma\delta \in cl_2(R_a \cup R_c). \tag{15}$$

Properties (14) and (15) together with our inductive hypothesis give:

$$\text{for any } q_t \in Q(T) \text{ such that } l \in q, q_t \in cl_2(R_a \cup R_c). \tag{16}$$

Furthermore (14) and (15) and the application of (2), first consequence, proves:

$$\alpha\beta|\gamma\delta \in cl_2(R_a \cup R_c). \tag{17}$$

Finally, let x, y and z be leaves of $A \cup B$. Let x be on a subtree of Δ, where y and z are not. By (14) (and with symmetry) we have $l\gamma|xy \in cl_2(R_a \cup R_c)$. Moreover, $lx|yz \in R_c$ due to our inductive hypothesis. These two together, through rule (2),

third consequence, give $x\gamma|yz \in cl_2(R_a \cup R_c)$. By symmetry, we also know the analogous result with subtrees T_Δ and T_Γ exchanged; therefore we have proved:

$$\text{if a quartet } q \in \binom{[n]}{4} \text{ contains three leaves from } T_\Delta, \text{ one leaf from } T_\Gamma, \text{ or vice versa, but } l \notin q, \text{ then } t_q \in cl_2(R_a \cup R_c). \tag{18}$$

Now, for the quartet $q = \{s, u, v, w\}$ such that every leaf $\in A \cup B \cup \{l\}$, it is easy to see that $q \in Q(T_\Gamma)$; therefore $t_q \in cl_2(p(T_\Gamma))$, and the "lifted up" version of this proof ensures that $t_q \in cl_2(R_c)$. This fact (and its analogues for the other half-graph) together with properties (16), (17) and (18) finish the proof of Case (b), and we are done. □

It is worth noting that Theorem 2 strengthens Theorem 1, as it is shown by the following result:

Lemma 3. For any semilabelled binary tree T on $[n], d(T) \leq \log_2 n - 1$.

Proof. Suppose edge e of T has maximal depth $d = d(T)$. Then there is a set V_e of at least 2^d vertices at distance $d - 1$ from e, and none of these can be a leaf of T. For $v \in V_e$ let $S(v)$ be set of leaves of T that become separated from e upon deletion of v. Since $S(v) \cap S(v') = \emptyset$ for $v \neq v'$, and $|S(v)| \geq 2$, we have $n = |\cup_{v \in V_e} S(v)| \geq 2|V_e| \geq 2 \times 2^d = 2^{d+1}$, as claimed. □

REFERENCES

[1] BANDELT, H.-J.—DRESS, A.: Reconstructing the shape of a tree from observed dissimilarity data. Advances in Applied Mathematics, Vol. 7, 1986, pp. 309–343.

[2] BRYANT, D. J.—STEEL, M. A.: Extension operations on sets of leaf-labelled trees. Advances in Applied Mathematics, Vol. 16, 1995, pp. 425–453.

[3] BUNEMAN, P.: The recovery of trees from measures of dissimilarity. In: Hodson, F. R., Kendall, D. G., Tautu, P. (Eds.): Mathematics in the Archaeological and Historical Sciences, Edinburgh University Press, Edinburgh 1971, pp. 387–395.

[4] COLONIUS, H.—SCHULTZE, H. H.: Tree structure for proximity data. Brit. J. Math. Stat. Psychol., Vol. 34, 1981, pp. 167–180 .

[5] DEKKER, M. C. H.: Reconstruction methods for derivation trees, Master's Thesis, Vrije Universiteit, Amsterdam 1986.

[6] KANNAN, S.,—LAWLER, E.—WARNOW, T: Determining the Evolutionary Tree Using Experiments. Journal of Algorithms Vol. 21, 1996, pp. 26-50.

[7] PEARL, J.—TARSI, M.: Structuring causal trees. J. Complexity, Vol. 2, 1986, pp. 60–77.

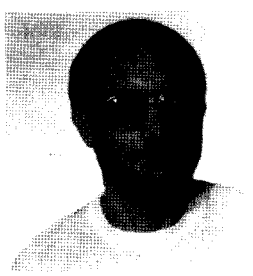
[8] PHILIPPE, H.—DOUZERY, E.: The pitfalls of molecular phylogeny based on four species, as illustrated by the cetacea/artiodactyla relationships. J. Mammal. Evol., Vol. 2, 1994, No. 2, pp. 133–152.

[9] STEEL, M. A.: The complexity of reconstructing trees from qualitative characters and subtrees. J. Classification, Vol. 9, 1992, pp. 91–116.

[10] STEEL, M. A.—SZÉKELY, L. A.—ERDŐS, P. L.: The number of nucleotide sites needed to accurately reconstruct large evolutionary trees, DIMACS, Rutgers University, New Jersey, USA 1996. DIMACS Technical Reports, pp. 96–43.

Péter L. ERDÖS received his Ph.D. degree from Eotvos University in 1982. His current affiliation is the Head of Department, Mathematical Institute of the Hungarian Academy of Sciences. Since 1990 he is a candidate of the Hungarian Academy of Sciences. He held the following positions: Assistant Professor of mathematics at the University of Economics in Budapest; Humboldt Fellow, University of Bonn, Institut für Ökonometrie ind Operationforschung, Germany; Associated Professor, Godollo University, Hungary. He also held different visiting positions at several universities in The Netherlands.

Mike A. STEEL, born in 1960, received his M.Sc. and Ph.D degrees in mathematics in 1983 from University of Canterbury and in 1989 from Massey University, respectively. At present he is the director of the Biomathematics Research Centre, senior lecturer at the Department of Mathematics and Statistics of the University of Canterbury (Christchurch, New Zealand).



László SZÉKELY received the Ph.D. in mathematics from Eotvos University, Budapest in 1983, and the Candidate of Mathematical Science degree from the Hungarian Academy of Sciences in 1987. He is Professor of Mathematics at the University of South Carolina since 1996. The positions held include: director of the Institute of Mathematics I at Eotvos University, Budapest (1994–96); Humboldt Fellow (1991–92); visiting professor at the University of New Mexico in Albuquerque (1988–90, 1992–93). His research area is in combinatorics and graph theory, with applications to computer science and biology.

Manuscript received 6 September 1996

Scientific Event

HPC 97
Fifth International Conference on Applications of High-Performance Computers in Engineering

2–4 July 1997, Centro de Supercomputación de Galicia, Santiago de Compostela, Spain

The objective of this Fifth International Conference on the Application of High-Performance Computing in Engineering is to bring together scientists and engineers working on the application of high-performance computers to solve complex engineering problems.
The application of supercomputing to numerical intensive problems brings several new issues that do not appear in standard computing. New algorithms and codes are required in order to exploit effectively the use of these novel computer architectures, as programs suitable for conventional computers are likely to achieve very modest improvement of performance on high-performance computers.
The field of high-performance computing is continuously changing. Although there are still changes being made in the hardware of high-performance computers, it is evident that the future of high-performance computing in engineering and science is in massively parallel computing. Therefore, engineers and scientists need to parallelise their numerical computer codes to be able to take advantage of the changes in high-performance computers. In that regard, the possible antagonism between algorithm and hardware, showing that there is a chance of spending too much time on programming different computer topologies versus think time, i.e. the time spent on the mathematical physics of the problem and numerical analysis in designing algorithms. Often a more natural mathematical algorithm founded on physical principles can lead to a better parallel computing formulation.
Conference topics: algorithms for parallelisation • distributed computer systems and networking • massively parallel systems • software tools and environments • performance and benchmarking • applications of neural computing • parallel finite and boundary elements • visualisation and graphics • applications in fluid flow • applications in structural mechanics • applications in applied science • transputer applications • distributed scheduling • industrial applications.

Contact address: Conference Secretariat
Paula Doughty-Young
HPC 97
Wessex Institute of Technology
Ashurst Lodge, Ashurst
Southampton, SO40 7AA
United Kingdom
Tel.: 44(0)1703 293223
Fax: 44(0)1703 292853
e-mail: Paula@wessex.ac.uk

A Few Logs Suffice to Build (Almost) All Trees (I)

Péter L. Erdős,[1] Michael A. Steel,[2] László A. Székely,[3] Tandy J. Warnow[4]
[1] *Mathematical Institute of the Hungarian Academy of Sciences, Budapest P.O. Box 127, Hungary-1364; e-mail: elp@math-inst.hu*
[2] *Biomathematics Research Centre, University of Canterbury, Christchurch, New Zealand; e-mail: m.steel@math.canterbury.ac.nz*
[3] *Department of Mathematics, University of South Carolina, Columbia, SC; e-mail: laszlo@math.sc.edu*
[4] *Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA; e-mail: tandy@central.cis.upenn.edu*

Received 26 September 1997; accepted 24 September 1998

ABSTRACT: A phylogenetic tree, also called an "evolutionary tree," is a leaf-labeled tree which represents the evolutionary history for a set of species, and the construction of such trees is a fundamental problem in biology. Here we address the issue of how many sequence sites are required in order to recover the tree with high probability when the sites evolve under standard Markov-style i.i.d. mutation models. We provide analytic upper and lower bounds for the required sequence length, by developing a new polynomial time algorithm. In particular, we show when the mutation probabilities are bounded the required sequence length can grow surprisingly slowly (a power of $\log n$) in the number n of sequences, for almost all trees. © 1999 John Wiley & Sons, Inc. Random Struct. Alg., 14, 153–184, 1999

1. INTRODUCTION

Rooted leaf-labeled trees are a convenient way to represent historical relationships between extant objects, particularly in evolutionary biology, where such trees are

Correspondence to: László A. Székely

called *phylogenies*. Molecular techniques have recently provided large amounts of sequence data which are being used to reconstruct such trees. These methods exploit the variation in the sequences due to random mutations that have occurred at the sites, and statistically based approaches typically assume that sites mutate independently and identically according to a Markov model. Under mild assumptions, for sequences generated by such a model, one can recover, with high probability, the underlying *unrooted* tree provided the sequences are sufficiently long in terms of the number k of sites. How large this value of k needs to be depends on the reconstruction method, the details of the model, and the number n of species. Determining bounds on k and its growth with n has become more pressing since biologists have begun to reconstruct trees on increasingly large numbers of species, often up to several hundred, from such sequences.

With this motivation, we provide upper and lower bounds for the value of k required to reconstruct an underlying (unrooted) tree with high probability, and address, in particular, the question of how fast k must grow with n. We first show that under any model, and any reconstruction method, k must grow *at least* as fast as $\log n$, and that for a particular, simple reconstruction method, it must grow at least as fast as $n \log n$, for any i.i.d. model. We then construct a new tree reconstruction method (the dyadic closure method) which, for a simple Markov model, provides an upper bound on k which depends only on n, the range of the mutation probabilities across the edges of the tree, and a quantity called the "depth" of the tree. We show that the depth grows very slowly ($O(\log \log n)$) for almost all phylogenetic trees (under two distributions on trees). As a consequence, we show that the value of k required for accurate tree reconstruction by the dyadic closure method needs only to grow as a power of $\log n$ for almost all trees when the mutation probabilities lie in a fixed interval, thereby improving results by Farach and Kannan in [23].

The structure of the paper is as follows. In Section 2 we provide definitions, and in Section 3 we provide lower bounds for k. In Section 4 we describe a technique for reconstructing a tree from a partial collection of subtrees, each on four leaves. We use this technique in Section 5, as the basis for our "dyadic closure" method. Section 6 is the central part of the paper, here we analyze, using various probabilistic arguments, an upper bound on the value of k required for this method to correctly recover the underlying tree with high probability, when the sites evolve under a simple, symmetric 2-state model. As this upper bound depends critically upon the depth (a function of the shape of the tree) we show that the depth grows very slowly ($O(\log \log n)$) for a random tree selected under either of two distributions. This gives us the result that k need grow only sublinearly in n for nearly all trees.

Our follow-up paper [21] extends the analysis presented in this paper for more general, r-state stochastic models, and offers an alternative to dyadic closure, the "witness–antiwitness" method. The witness–antiwitness method is faster than the dyadic closure method on average, but does not yield a deterministic technique for reconstructing a tree from a partial collection of subtrees, as the dyadic closure method does; furthermore, the witness–antiwitness method may require somewhat longer (by a constant multiplicative factor) input sequences than the dyadic closure method.

2. DEFINITIONS

Notation. $\mathbb{P}[A]$ denotes the probability of event A; $\mathbb{E}[X]$ denotes the expectation of random variable X. We denote the natural logarithm by log. The set $[n]$ denotes $\{1, 2, \ldots, n\}$ and for any set S, $\binom{S}{k}$ denotes the collection of subsets of S of size k. $\mathbb{R}$ denotes the real numbers.

Definitions. (I) Trees. We will represent a phylogenetic tree T by a tree whose *leaves* (vertices of degree 1) are labeled (by extant species, numbered by $1, 2, \ldots, n$) and whose remaining internal vertices (representing ancestral species) are unlabeled. We will adopt the biological convention that phylogenetic trees are *binary*, so that all internal nodes have degree 3, and we will also assume that T is *unrooted*, for reasons described later in this section. There are $(2n-5)!! = (2n-5)(2n-7)\cdots 3\cdot 1$ different binary trees on n distinctly labeled leaves.

The edge set of the tree is denoted by $E(T)$. Any edge adjacent to a leaf is called a *leaf edge*, any other edge is called an *internal edge*. The path between the vertices u and v in the tree is called the uv path, and is denoted $P(u, v)$. For a phylogenetic tree T and $S \subseteq [n]$, there is a unique minimal subtree of T, containing all elements of S. We call this tree the *subtree* of T induced by S, and denote it by $T_{|S}$. We obtain the *contracted subtree* induced by S, denoted by $T^*_{|S}$, if we substitute edges for all maximal paths of $T_{|S}$ in which every internal vertex has degree 2. Since all trees are assumed to be binary, all contracted subtrees, including, in particular, the subtrees on four leaves, are also binary. We use the notation $ij|kl$ for the contracted subtree on four leaves i, j, k, l in which the pair i, j is separated from the pair k, l by an internal edge, and we also call $ij|kl$ a *valid quartet split* of T. Clearly any four leaves i, j, k, l in a binary tree have exactly one valid quartet split out of $ij|kl, ik|jl, il|kj$.

The *topological distance* $d(u, v)$ between vertices u and v in a tree T is the number of edges in $P(u, v)$. A *cherry* in a binary tree is a pair of leaves at topological distance 2. The *diameter* of the tree T, $\mathrm{diam}(T)$, is the maximum topological distance in the tree. For an edge e of T, let T_1 and T_2 be the two rooted subtrees of T obtained by deleting edge e from T, and for $i = 1, 2$, let $d_i(e)$ be the topological distance from the root of T_i to its nearest leaf in T_i. The *depth* of T is $\max_e \max\{d_1(e), d_2(e)\}$, where e ranges over all internal edges in T. We say that a path P in the tree T is *short* if its topological length is at most $\mathrm{depth}(T) + 1$, and say that a quartet i, j, k, l is a *short quartet* if it induces a subtree which contains a single edge connected to four disjoint short paths. The set of all short quartets of the tree T is denoted by $Q_{\mathrm{short}}(T)$. We will denote the set of valid quartet splits for the short quartets by $Q^*_{\mathrm{short}}(T)$.

(II) Sites. Let us be given a set C of character states (such as $C = \{A, C, G, T\}$ for DNA sequences; $C = \{\text{the 20 amino acids}\}$ for protein sequences; $C = \{R, Y\}$ or $\{0, 1\}$ for purine-pyrimidine sequences). A *sequence of length k* is an ordered k-tuple from C—that is, an element of C^k. A collection of n such sequences—one for each species labeled from $[n]$—is called a *collection of aligned sequences*.

Aligned sequences have a convenient alternative description as follows. Place the aligned sequences as rows of an $n \times k$ matrix, and call *site* i the ith column of this matrix. A *pattern* is one of the $|C|^n$ possible columns.

(III) Site substitution models. Many models have been proposed to describe, stochastically, the evolution of sites. Usually these models assume that the sites evolve identically and independently under a distribution that depends on the model tree. Most models are more specific and also assume that each site evolves on a rooted tree from a nondegenerate distribution π of the r possible states at the root, according to a Markov assumption (namely, that the state at each vertex is dependent only on its immediate parent). Each edge e oriented out from the root has an associated $r \times r$ stochastic transition matrix $M(e)$. Although these models are usually defined on a rooted binary tree T where the orientation is provided by a time scale and the root has degree 2, these models can equally well be described on an unrooted binary tree by (i) suppressing the degree 2 vertex in T, (ii) selecting an arbitrary vertex (leaves not excluded), assigning to it an appropriate distribution of states π', possibly different from π, and (iii) assigning an appropriate transition matrix $M'(e)$ [possibly different from $M(e)$] for each edge e. If we regard the tree as now rooted at the selected vertex, and the "appropriate" choices in (ii) and (iii) are made, then the resulting models give *exactly* the same distribution on patterns as the original model (see [46]) and as the rerooting is arbitrary we see why it is impossible to hope for the reconstruction of more than the *unrooted* underlying tree that generated the sequences under some time-induced, edge-bisection rooting. The assumption that the underlying tree is binary is also in keeping with the assumption in systematic biology, that speciation events are almost always binary.

(IV) The Neyman model. The simplest stochastic model is a symmetric model for binary characters due to Neyman [37], and also developed independently by Cavender [12] and Farris [25]. Let $\{0, 1\}$ denote the two states. The root is a fixed leaf, the distribution π at the root is uniform. For each edge e of T we have an associated *mutation probability*, which lies strictly between 0 and 0.5. Let p: $E(T) \to (0, 0.5)$ denote the associated map. We have an instance of the general Markov model with $M(e)_{01} = M(e)_{10} = p(e)$. We will call this the *Neyman 2-state model*, but note that it has also been called the Cavender–Farris model. Neyman's original paper allows more than 2 states.

The Neyman 2-state model is hereditary on the subsets of the leaves—that is, if we select a subset S of $[n]$, and form the subtree $T_{|S}$, then eliminate vertices of degree 2, we can define mutation probabilities on the edges of $T^*_{|S}$ so that the probability distribution on the patterns on S is the same as the marginal of the distribution on patterns provided by the original tree T. Furthermore, the mutation probabilities that we assign to an edge of $T^*_{|S}$ is just the probability p that the endpoints of the associated path in the original tree T are in different states. The probability that the endpoints of a path p are in different states is nicely related to the mutation probabilities $p_1, p_2, \ldots, p_k$ of edges of the k-path,

$$p = \frac{1}{2}\left(1 - \prod_{i=1}^{k}(1 - 2p_i)\right). \tag{1}$$

Formula (1) is well known, and is easy to prove by induction.

(V) Distances. Any symmetric matrix, which is zero-diagonal and positive off-diagonal, will be called a *distance matrix*. An $n \times n$ distance matrix D_{ij} is called *additive*, if there exists an n-leaf (not necessarily binary) with positive edge weights on the internal edges and nonnegative edge weights on the leaf edges, so that D_{ij} equals the sum of edge weights in the tree along the $P(i, j)$ path connecting i and j. In [10], Buneman showed that the following Four-Point Condition characterizes additive matrices (see also [42] and [53]):

Theorem 1 (Four-Point Condition). A matrix D is additive if and only if for all i, j, k, l (not necessarily distinct), the maximum of $D_{ij} + D_{kl}, D_{ik} + D_{jl}, D_{il} + D_{jk}$ is not unique. The edge-weighted tree with positive weights on internal edges and nonnegative weights on leaf edges representing the additive distance matrix is unique among the trees without vertices of degree 2.

Given a pair of parameters (T, p) for the Neyman 2-state model, and sequences of length k generated by the model, let $H(i, j)$ denote the *Hamming distance* of sequences i and j and

$$h^{ij} = \frac{H(i, j)}{k} \tag{2}$$

denote the *dissimilarity score* of sequences i and j. The *empirical corrected distance* between i and j is denoted by

$$d_{ij} = -\tfrac{1}{2}\log(1 - 2h^{ij}). \tag{3}$$

The probability of a change in the state of any fixed character between the sequences i and j is denoted by $E^{ij} = \mathbb{E}(h^{ij})$, and we let

$$D_{ij} = -\tfrac{1}{2}\log(1 - 2E^{ij}) \tag{4}$$

denote the *corrected model distance* between i and j. We assign to any edge e a positive weight,

$$w(e) = -\tfrac{1}{2}\log(1 - 2p(e)). \tag{5}$$

By Eq. (1), D_{ij} is the sum of the weights (see previous equation) along the path $P(i, j)$ between i and j. Therefore, d_{ij} converges in probability to D_{ij} as $k \to \infty$. Corrected distances were introduced to handle the problem that Hamming distances underestimate the "true evolutionary distances." In certain continuous time Markov models the edge weight means the expected number of back-and-forth state changes along the edge, and defines an additive distance matrix.

(VI) Tree reconstruction. A *phylogenetic tree reconstruction method* is a function Φ that associates either a tree or the statement `fail` to every collection of aligned sequences, the latter indicating that the method is unable to make such a selection for the data given. Some methods are based upon sequences, while others are based upon distances.

According to the practice in systematic biology (see, for example, [29, 30, 49]), a method is considered to be *accurate* if it recovers the unrooted binary tree T, even if it does not provide any estimate of the mutation probabilities. A necessary condition for accuracy, under the models discussed above, is that two distinct trees, T, T', do not produce the same distribution of patterns no matter how the trees are rooted, and no matter what their underlying Markov parameters are. This "identifiability" condition is violated under an extension of the i.i.d. Markov model when there is an unknown distribution of rates across sites as described by Steel, Székely, and Hendy [46]. However, it is shown in Steel [44] (see also Chang and Hartigan [13]) that the identifiability condition holds for the i.i.d. model under the weak conditions that the components of π are not zero and the determinant $\det(M(e)) \neq 0, 1, -1$, and in fact we can recover the underlying tree from the expected frequencies of patterns on just *pairs* of species.

Theorem 1 and the discussion that follows it suggest that appropriate methods applied to corrected distances will recover the correct tree topology from sufficiently long sequences. Consequently, one approach to reconstructing trees from distances is to seek an additive distance matrix of minimum distance (with respect to some metric on distance matrices) from the input distance matrix. Many metrics have been considered, but all resultant optimization problems have been shown or are assumed to be NP-hard; see [1, 15, 24].

We will use a particular simple distance method, which we call the (*Extended Four-Point Method* (FPM), to reconstruct trees on four leaves from a matrix of interleaf distances.

Four-Point Method (*FPM*). *Given a* 4×4 *distance matrix* d, *return the set of splits* $ij|kl$ *which satisfy* $d_{ij} + d_{kl} \leq \min\{d_{ik} + d_{jl}, d_{il} + d_{jk}\}$.

Note that the Four-Point Method can return one, two, or three splits for a given quartet. One split is returned if the minimum is unique, two are returned if the two smallest values are identical but smaller than the largest, and three are returned if all three values are equal.

In [26], Felsenstein showed that two popular methods—*maximum parsimony* and *maximum compatibility*—can be statistically inconsistent, namely, for some parameters of the model, the probability of recovering the correct tree topology tends to 0 as the sequence length grows. This region of the parameter space has been subsequently named the "Felsenstein zone." This result, and other more recent embellishments (see Hendy [28], Zharkikh and Li [54], Takezaki and Nei [50], Steel, Székely, and Hendy [46]), are asymptotic results—that is, they are concerned with outcomes as the sequence length, k, tends to infinity.

We consider the question of how many sites k must be generated independently and identically, according to a substitution model M, in order to reconstruct the underlying binary tree on n species with prespecified probability at least ϵ by a particular method Φ. Clearly, the answer will depend on Φ, ϵ, and n, and also on the fine details of M—in particular the unknown values of its parameters. It is clear that for all models that have been proposed, if no restrictions are placed on the parameters associated with edges of the tree then the sequence length might need to be astronomically large, even for four sequences, since the "edge length" of the internal edge(s) of the tree can be made arbitrarily short (as was pointed out by Philippe and Douzery [38]). A similar problem arises for four sequences when one or more of the four noninternal edges is "long"—that is, when site saturation

has occurred on the line of descent represented by the edge(s). Unfortunately, it is difficult to analyze how well methods perform for sequences of a given length, k. There has been some empirical work done on this subject, in which simulations of sequences are made on different trees and different methods compared according to the sequence length needed (see [31] for an example of a particularly interesting study of sequence length needed to infer trees of size 4), but little analytical work (see, however, [38]).

In this paper we consider only the Neyman 2-state model as our choice for M. However, our results extend to the general i.i.d. Markov model, and the interested reader is referred to the companion paper [21] for details.

3. LOWER BOUNDS

Since the number of binary trees on n leaves is $(2n-5)!!$, encoding deterministically all such trees by binary sequences at the leaves requires that the sequence length, k, satisfy $(2n-5)!! \le 2^{nk}$, i.e., $k = \Omega(\log n)$. We now show that this information-theoretic argument can be extended for *arbitrary* models of site evolution and *arbitrary* deterministic or even randomized algorithms for tree reconstruction. For each tree, T, and for each algorithm A, whether deterministic or randomized, we will assume that T is equipped with a mechanism for generating sequences, which allows the algorithm A to reconstruct the topology of the underlying tree T from the sequences with probability bounded from below.

Theorem 2. *Let A be an arbitrary algorithm, deterministic or randomized, which is used to reconstruct binary trees from* 0-1 *sequences of length k associated with the leaves, under an arbitrary model of substitutions. If A reconstructs the topology of any binary tree T from the sequences at the leaves with probability greater than ϵ (respectively, greater than $\frac{1}{2}$), then $(2n-5)!!\,\epsilon < 2^{nk}$ (respectively, $(2n-5)!! \le 2^{nk}$, under the assumption of (stochastic) independence of the substitution model and the reconstruction) and so $k = \Omega(\log n)$.*

We prove this theorem in a more abstract setting:

Theorem 3. *We have finite sets X and S and random functions $f\colon S \to X$ and $g\colon X \to S$.*

(i) *If $\mathbb{P}[fg(x) = x] > \epsilon$ for all $x \in X$ then $|S| > \epsilon |X|$.*
(ii) *If f, g are independent and $\mathbb{P}[fg(x) = x] > \frac{1}{2}$ for all $x \in X$ then $|S| \ge |X|$.*

Proof. Proof of (i). By hypothesis $\epsilon|X| < \sum_x \mathbb{P}[fg(x) = x] = \sum_x \sum_s \mathbb{P}[g(x) = s$ and $f(s) = x] \le \sum_s (\sum_x \mathbb{P}[f(s) = x]) = \sum_s 1 = |S|$.

Proof of (ii). First note that $\mathbb{P}[fg(x) = y] = \sum_s \mathbb{P}[f(s) = y]\mathbb{P}[g(x) = s]$ by independence. Observe that for each x, there exists an $s = s_x$ for which $\mathbb{P}[f(s_x) = x] > \frac{1}{2}$, since otherwise we have $\mathbb{P}[fg(x) = x] \le \frac{1}{2}$. Now, the map sending x to s_x is one-to-one from X into S (and so $|X| \le |S|$ as required) since otherwise, if two elements get mapped to s, then $1 = \sum_x \mathbb{P}[f(s) = x] > \frac{1}{2} + \frac{1}{2}$. ∎

The following example shows that our theorem is tight for $\epsilon < \frac{1}{2}$: Let $X = \{x_{11}, x_{12}, x_{21}, x_{22}, \ldots, x_{n1}, x_{n2}\}$ and $S = \{1, 2, \ldots, n\}$, and let $g(x_{ij}) = i$ (with probability 1); and let $f(i) = x_{i1}$ with probability $\frac{1}{2}$; x_{i2} with probability $\frac{1}{2}$. Then $\mathbb{P}[fg(x) = x] = \frac{1}{2}$, so $\mathbb{P}[fg(x) = x] > \epsilon$, for *any* epsilon less than $\frac{1}{2}$. However, notice that $|X|/2 = |S|$.

Curiously, once ϵ exceeds $\frac{1}{2}$ we must have $|X| \leq |S|$, under the assumption of independence. Examples [52] show that the assumption of independence is necessary. Independence is a reasonable assumption if we try to apply this result for evolutionary tree reconstruction, and holds automatically if the tree reconstruction method is deterministic.

This lower bound applied to an *arbitrary* algorithm, but *particular* algorithms may admit much larger lower bounds. Consider, for example, the *Maximum Compatibility Method* (MC), which we now define. Given a set of binary sequences, each site defines a partition of the sequences into two sets, those containing a 0 in that position, and those containing a 1 in that position. The site is said to be *compatible* on a tree T if the tree T contains an edge whose removal would define the same partition. The objective of the maximum compatibility method is a tree T which has the largest number of sites compatible with it. Maximum compatibility is an NP-hard optimization problem [16], although the MC method can clearly be implemented as a nonpolynomial time algorithm. We now show that the sequence length needed by MC to obtain the correct topology with constant probability must grow *at least* as fast as $n \log n$.

Theorem 4. *Assume that 2-state sites on n species evolve on a binary tree T according to any stochastic model in which the sites evolve identically and independently. Let $k(n)$ denote the smallest number of sites for which the Maximum Compatibility Method is guaranteed to reconstruct the topology of T with probability greater than $\frac{1}{2}$. Then, for n large enough,*

$$k(n) > (n-3)\log(n-3) - (n-3). \tag{6}$$

Proof. We say that a site is *trivial* if it defines a partition of the sequences into one class or into two classes so that one of the classes is a singleton. Now, fix x and assume that we are given $k^* = \lceil (n-3)\log(n-3) + x(n-3) \rceil$ nontrivial sites independently selected from the same distribution. We show that the probability of obtaining the correct tree under MC is at most $e^{-e^{-x}}$ for n large enough. This proves the theorem by setting $x = -1$, since $k(n) \geq k^*|_{x=-1}$ is needed.

Let $\sigma(T)$ denote the set of internal splits of T. Since T is binary, $|\sigma(T)| = n - 3$ [10]. For $\sigma \in \sigma(T)$, let the random variable X_σ be the number of nontrivial sites which induce split σ. Define $X = \sum_{\sigma \in \sigma(T)} X_\sigma$. A necessary (though not sufficient) condition for maximum compatibility to select T is that all the internal splits of T are present among the k^* nontrivial sites. Thus, we have the inequality,

$$\begin{aligned}
\mathbb{P}[MC(\mathbf{S}) = T] &\leq \mathbb{P}\Big[\bigcap_{\sigma \in \sigma(T)} \{X_\sigma > 0\}\Big] \\
&= \sum_{i=1}^{k^*} \mathbb{P}\Big[\bigcap_{\sigma \in \sigma(T)} \{X_\sigma > 0\} | X = i\Big] \times \mathbb{P}[X = i] \\
&\leq \max_{1 \leq i \leq k^*} \mathbb{P}\Big[\bigcap_{\sigma \in \sigma(T)} \{X_\sigma > 0\} | X = i\Big] \\
&= \mathbb{P}\Big[\bigcap_{\sigma \in \sigma(T)} \{X_\sigma > 0\} | X = k^*\Big]. \qquad (7)
\end{aligned}$$

Let $p(\sigma)$ denote the probability of generating split σ at a particular site. Due to the model, $p(\sigma)$ does not depend on the site. It is not difficult to show that (7) is maximized when the $p(\sigma)$s are all equal ($\sigma \in \sigma(T)$) and sum to 1.

Indeed, by compactness arguments, there exists a probability distribution maximizing (7). We show that it cannot be nonuniform, and therefore the uniform distribution maximizes (7). Assume that the maximizing distribution p is nonuniform, say, $p(\sigma) \neq p(\rho)$. We introduce a new distribution p' with $p'(\sigma) = p'(\rho) = \frac{1}{2}(p(\sigma) + p(\rho))$, and $p'(\alpha) = p(\alpha)$ for $\alpha \neq \sigma, \rho$. The probability of having exactly i sites supporting σ or ρ is the same for p and p'. Conditioning on the number of sites supporting σ or ρ, it is easy to see that any distribution of sites supporting all nontrivial splits has strictly higher probability in p' than in p.

Knowing that the $p(\sigma)$s are all equal ($\sigma \in \sigma(T)$) and sum to 1, determining (7) is just the classical occupancy problem where k^* balls are randomly assigned to $n-3$ boxes with uniform distribution, and one asks for the probability that each box has at least one ball in it. Equation (6) now follows from a result on the asymptotics of this problem (Erdős and Rényi [18]): for $x \in \mathbb{R}$, k^* balls (k^* as defined above), and $n-3$ boxes, the limit of probability of filling each boxes is $e^{-e^{-x}}$. ■

This theorem shows that the sequence length that suffices for the MC method to be accurate is in $\Omega(n \log n)$, but does not provide us with any *upper bound* on that sequence length. This upper bound remains an open problem.

In Section 5, we will present a new method [the *Dyadic Closure Method* (DCM)] for reconstructing trees. DCM has the property that for almost all trees, with a wide range allowed for the mutation probabilities, the sequence length that *suffices* for correct topology reconstruction grows no more than polynomially in the lower bound of $\log n$ (see Theorem 2) required for any method. In fact the same holds for *all trees* with a *narrow range* allowed for the mutation probabilities. First, however, we set up a combinatorial technique for reconstructing trees from selected subtrees of size 4.

4. DYADIC INFERENCE OF TREES

Certain classical tree reconstruction methods [6, 14, 47, 48, 55] are based upon reconstructing trees on quartets of leaves, them combining these trees into one tree on the entire set of leaves. Here we describe a method which requires only certain quartet splits be reconstructed (the "representative quartet splits"), and then infers the remaining quartet splits using "inference rules." Once we have splits for all the possible quartets of leaves, we can then reconstruct the tree (if one exists) that is uniquely consistent with all the quartet splits.

In this section, we prove a stronger result than was provided in [19], that the *representative quartet splits* suffice to define the tree. We also present a tree reconstruction algorithm, DCTC (for *Dyadic Closure Tree Construction*) based upon dyadic closure. The input to DCTC is a set Q of quartet splits and we show that DCTC is guaranteed to reconstruct the tree properly if the set Q contains only valid quartet splits and contains all the representative quartet splits of T. We also show that if Q contains all representative quartet splits but also contains invalid

quartet splits, then DCTC discovers incompatibility. In the remaining case, where Q does not contain all the representative quartet splits of any T, DCTC returns *Inconsistent* (and then the input was inconsistent indeed), or a tree (which is then the only tree consistent with the input), or *Insufficient*.

4.1. Inference Rules

Recall that, for a binary tree T on n leaves, and a quartet of leaves,

$$q = \{a, b, c, d\} \in \binom{[n]}{4}, \qquad t_q = ab|cd$$

is a *valid quartet split* of T if $T^*_{|q} = ab|cd$ (i.e., there is at least one edge in T whose removal separates the pair a, b from the pair c, d). It is easy to see that

$$\text{if } ab|cd \text{ is a valid quartet split of T, then so are } ba|cd \text{ and } cd|ab, \tag{8}$$

and we identify these three splits; and if $ab|cd$ holds, then $ac|bd$ and $ad|bc$ are not valid quartet splits of T, and we say that any of them *contradicts* $ab|cd$. Let

$$Q(T) = \left\{ t_q \colon q \in \binom{[n]}{4} \right\}$$

denote the set of valid quartet splits of T. It is a classical result that $Q(T)$ determines T (Colonius and Schulze [14], Bandelt and Dress [6]); indeed for each $i \in [n]$, $\{t_q \colon i \in q\}$ determines T, and T can be computed from $\{t_q \colon i \in q\}$ in polynomial time.

It would be nice to determine for a set of quartet splits whether there is a tree for which they are valid quartet splits. Unfortunately, this problem is NP-complete (Steel [43]). It also would be useful to know which subsets of $Q(T)$ determine T, and for which subsets a polynomial time procedure would exist to reconstruct T. A natural step in this direction is to define *inference*: we can infer from a set of quartet splits A a quartet split t, if whenever $A \subseteq Q(T)$ for a binary tree T, then $t \in Q(T)$ as well.

Instead, Dekker [17] introduced a restricted concept, *dyadic* and higher order inference. Following Dekker, we say that a set of quartet splits A *dyadically implies* a quartet split t, if t can be derived from A by repeated applications of rules (8)–(10):

$$\begin{gathered}\text{if } ab|cd \text{ and } ac|de \text{ are valid quartet splits of } T,\\ \text{then so are } ab|ce,\ ab|de, \text{ and } bc|de,\end{gathered} \tag{9}$$

and,

$$\text{if } ab|cd \text{ and } ab|ce \text{ are valid quartet splits of } T, \text{ then so is } ab|de. \tag{10}$$

It is easy to check that these rules infer valid quartet splits from valid quartet splits, and the set of quartet splits dyadically inferred from an input set of quartet splits can be computed in polynomial time. Setting a complete list of inference rules seems hopeless (Bryant and Steel [9]): for any r, there are r-ary inference rules,

which infer a valid quartet split from some r valid quartet splits, such that their action cannot be expressed through lower order inference rules.

4.2. Tree Inference Using Dyadic Rules

In this section we define the *dyadic closure* of a set of quartet splits, and describe conditions on the set of quartet splits under which the dyadic closure defines all valid quartet splits of a binary tree. This section extends and strengthens results from earlier work [19, 45].

Definition 1. Given a finite set of quartet splits Q, we define the *dyadic closure* $\mathrm{cl}(Q)$ of Q as the set of quartet splits than can be inferred from Q by the repeated use of the rules (8–10). We say that Q is *inconsistent*, if Q is not contained in the set of valid quartet splits of any tree, otherwise Q is *consistent*. For each of the $n-3$ internal edges of the n-leaf binary tree T we assign a *representative quartet* $\{s_1, s_2, s_3, s_4\}$ as follows. The deletion of the internal edge and its endpoints defines four rooted subtrees t_1, t_2, t_3, t_4. Within each subtree t_i, select from among the leaves which are closest topologically to the root the one, s_i, which is the smallest natural number (recall that the leaves of our trees are natural numbers). This procedure associates to each edge a set of four leaves, i, j, k, l. (By construction, it is clear that the quartet i, j, k, l induces a short quartet in T—see Section 2 for the definition of "short quartet.") We call the quartet split of a representative quartet a *representative quartet split* of T, and we denote the set of representative quartet splits of T by R_T.

The aim of this section is to show that the dyadic closure suffices to compute the tree T from any set of valid quartet splits of T which contain R_T. We begin with:

Lemma 1. *Suppose S is a set of $n-3$ quartet splits which is consistent with a unique binary tree T on n leaves. Furthermore, suppose that S can be ordered $q_1, \ldots, q_{n-3}$ in such a way that q_i contains at least one label which does not appear in $\{q_1, \ldots, q_{i-1}\}$ for $i = 2, \ldots, n-3$. Then, the dyadic closure of S is $Q(T)$.*

Proof. First, observe that it is sufficient to show the lemma for the case when q_i contains *exactly* one label which does not appear in $\{q_1, \ldots, q_{i-1}\}$ for $i = 2, \ldots, n-3$, since $n-4$ quartets have to add $n-4$ new vertices. Let $S_i = \{q_1, \ldots, q_i\}$, and let L_i be the union of the leaves of the quartet splits in S_i, and let $T_i = T^*_{|L_i}$ be the binary subtree of T induced by L_i. We first make

Claim 1. *The only tree on L_i consistent with S_i is T_i, for $1, \ldots, n-3$.*

Proof of Claim 1. The claim is true by the hypothesis of Lemma 1 for $i = n-3$; suppose for some $i < n-3$ it is false. Then there exist (at least) two trees that realize S_i, one of which is T_i, the other we will call $T^{\#}$. Now each quartet $q_{i+1}, \ldots, q_{n-3}$ adds a new leaf to the tree so far constructed from T_i and $T^{\#}$. Now for each quartet *we can always* attach that new leaf in at least one position in the tree so far constructed so as to satisfy the corresponding quartet split (and all earlier ones, since they don't involve that leaf). Thus we end up with two trees consistent with S, and these are different trees since when we restrict them to L_i, they differ. But this contradicts our hypothesis. ∎

Next we make

Claim 2. *If x is the new leaf introduced by $q_{n-3} = xa|bc$ then x and a form a cherry of T.*

Proof of Claim 2. First assume that x belongs to the cherry xy but $a \neq y$. Since this quartet is the only occurrence of x we do not have any information about this cherry, therefore the reconstruction of the tree T cannot be correct, a contradiction.

Now assume that x is not in a cherry at all. Then the neighbor of x has two other neighbors, and those are not leaves. In turn they have two other neighbors each. Hence, we can describe x's place in T in the following representation in Fig. 1: take a binary tree with five leaves, label the middle leaf x, and replace the other four leaves by corresponding subtrees of T.

Now suppose $q_{n-3} = ax|bc$. Regardless of where a, b, c come from (among the four subtrees in the representation), we can always move x onto at least two of the other four edges in T, and so obtain a different tree consistent with S (recall that q_{n-3} is the only quartet containing x, and thereby the only obstruction to us moving x!). Since the theorem assumes that the quartets are consistent with a unique tree, this contradicts our assumptions. ■

Finally, it is easy to show the following:

Claim 3. *Suppose xy is a cherry of T. Select leaves a, b from each of the two subtrees adjacent to the cherry. Let T' be the binary tree obtained by deleting leaf x. Then* $\mathrm{cl}(Q(T') \cup \{xy|ab\}) = Q(T)$.

Now, we can apply induction on n to establish the lemma. It is clearly (vacuously) true for $n = 4$, so suppose $n > 4$. Let x be the new leaf introduced by q_{n-3}, and let the binary tree T' be T with x deleted.

In view of Claim 1, S_{n-4} is a set of $n-4$ quartets that define $T_{n-4} = T'$, a tree on $n-1$ leaves and which satisfy the hypothesis that q_i introduces exactly one new leaf. Thus, applying the induction hypothesis, the dyadic closure of S_{n-4} is $Q(T')$. Since $S = S_{n-3}$ contains S_{n-4}, the dyadic closure of S also contains $Q(T')$, which is the set of all quartet splits of T that do not include x.

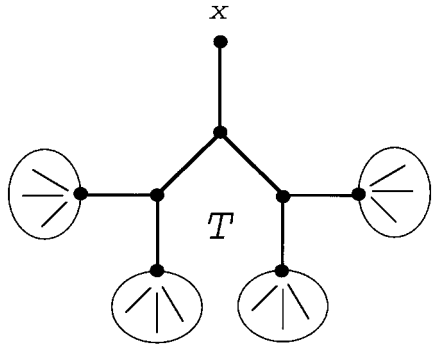


Fig. 1. Position of a leaf x, which is not a cherry, in a binary tree.

Now, by Claim 2, x is in a cherry; let its sibling in the cherry be y, so $q_{n-3} = ab|xy$, say, where a and b must lie in each of the two subtrees adjacent to the cherry. (It is easy to see that if a, b both lie in just one of these subtrees, then S would not define T.)

Now, as we just said, the dyadic closure of S contains $Q(T')$ and it also contains $ab|xy$ (where a, b are as specified in the preceding paragraph) and so by the idempotent nature of dyadic closure [i.e., $\mathrm{cl}(B) = \mathrm{cl}(\mathrm{cl}(B))$] it follows from Claim 3 that the dyadic closure of S equals $Q(T)$. ■ ■ ■

Lemma 2. *The set of representative quartet splits R_T of a binary tree T satisfies the conditions of Lemma* 1. *Hence, the dyadic closure of R_T is $Q(T)$.*

Proof. In order to make an induction proof possible, we make a more general statement. Given a binary tree T with a positive edge weighting w, we define the *representative quartet* of an edge e to be the quartet tree defined by taking the lowest indiced closest leaf in each of the four subtrees, where we define "closest" in terms of the weight of the path (rather than the topological distance) to the root of the subtree. We also define the *representative quartet splits of the weighted tree*, $R_{T,w}$ as in the definition of representative quartets of unweighted trees, with the only change being that each $s_i \in t_i$ is selected to minimize the *weighted* path length rather than topological path length (i.e., the edge weights on the path are summed together, to compute the weighted path length). Observe that if all weights are equal to 1, then we get back the original definitions. When turning to binary subtrees of a given weighted tree, we assign the sum of weights of the original edges to any newly created edge which is composed of them, and denote the new weighting by w^*. Now we can easily prove by induction the following generalization of the statement of Lemma 2:

Claim 4. *Take the set of representative quartet splits $R_{T,w}$ of a weighted n-leaf binary tree T. Then for every other n-leaf binary tree F, we have that $R_{T,w} \subseteq Q(F)$ implies $T = F$ as unweighted trees. Furthermore, $R_{T,w}$ can be ordered $q_1, \ldots, q_{n-3}$ in such a way that q_i contains exactly one label that does not appear in $\{q_1, \ldots, q_{i-1}\}$ for $i = 2, \ldots, n-3$.*

Proof of Claim 4. First we show that the only tree consistent with the set of representative splits $R_{T,w}$ of a binary tree T is T itself. Look for the smallest (in n) counterexample T, such that $R_{T,w} \subseteq Q(F)$ for a tree $F \neq T$. Clearly n has to be at least 5. Therefore T has at least two different cherries, say xy and uv, such that $d(u, x) \geq 4$. Let us denote by $w(l)$ the weight of the leaf edge corresponding to the leaf l. If $w(x) < w(y)$ or $[w(x) = w(y)$ and $x < y]$, then due to the construction of $R_{T,w}$, vertex y occurs in exactly one elements of $R_{T,w}$, say p, which is the representative of the edge that separates xy from the rest of the tree. A similar argument would show that one of u, v, say v, occurs in exactly one element of $R_{T,w}$, say q. It also follows that $p \neq q$. It is not difficult to check that

$$R_{T^*_{[[n]\setminus\{y\}}, w^*}} = R_T \setminus \{p\} \quad \text{and} \quad R_{T^*_{[[n]\setminus\{v\}}w^*}} = R_T \setminus \{q\} \tag{11}$$

according to the definition of weight after contracting edges, where $T^*_{|K}$ is the binary tree obtained by contracting paths into edges in the subtree of T spanned by the vertex set K. Hence, by the minimality of the counterexample, $T^*_{|[n]\setminus\{y\}} = F^*_{|[n]\setminus\{y\}}$ and $T^*_{|[n]\setminus\{v\}} = F^*_{|[n]\setminus\{v\}}$. We know that any edge of F defines a bipartition of $[n]$, and traces of these bipartitions on $[n]\setminus\{y\}$ and $[n]\setminus\{v\}$ are exactly the bipartitions produced by the edges of $F^*_{|[n]\setminus\{y\}}$ on $[n]\setminus\{y\}$ and the bipartitions produced by the edges of $F^*_{|[n]\setminus\{v\}}$ on $[n]\setminus\{v\}$. Therefore also in F both xy and uv make cherries, and hence $T = F$, a contradiction.

For the other part of the claim, it immediately follows by induction from formula (11) that $R_{T,w}$ can be ordered so that every quartet in the order contains *at least one* (and therefore *exactly one*) new leaf. [Eliminate quartet splits recursively using (11), and put $R_{T,w}$ in the reverse order.] ■

Note that the generalization for weighted trees was necessary, since without weights formula (11) would fail. ■ ■ ■

We note here that representative quartets cannot be defined by selecting *any* nearest leaf in the four subtrees associated with an internal edge. For example, consider the tree T on six leaves labeled 1 through 6, with a central vertex and cherries $(1,2)$, $(3,4)$, and $(5,6)$, hanging from the central vertex. If we selected the quartet splits by arbitrarily picking closest leaves in each of the four subtrees around each internal edge, we could possibly select splits $12|36$, $34|15$, and $56|24$; however, these splits do not uniquely identify the tree T, since the tree with cherries 15, 24, and 36, is also consistent with these quartets.

4.3. Dyadic Closure Tree Construction Algorithm

We now present the Dyadic Closure Tree Construction method (DCTC) for computing the dyadic closure of a set Q of quartet splits, and which returns the tree T when $\mathrm{cl}(Q) = Q(T)$.

Before we present the algorithm, we note the following interesting lemma:

Lemma 3. *If* $\mathrm{cl}(Q)$ *contains exactly one split for each possible quartet then* $\mathrm{cl}(Q) = Q(T)$ *for a unique binary tree* T.

Proof. By Proposition (2) of [6], a set Q^* of noncontradictory quartet splits equals $Q(T)$ for some tree T precisely if it satisfies the substitution property: If $ab|cd \in Q^*$, then for all $e \notin \{a,b,c,d\}$, $ab|ce \in Q^*$, or $ae|cd \in Q^*$. Furthermore, in that case, T is unique.

Applying this characterization to $Q^* = \mathrm{cl}(Q)$, suppose $ab|cd \in \mathrm{cl}(Q)$ but $ab|ce \notin \mathrm{cl}(Q)$. Thus, either $ae|bc \in \mathrm{cl}(Q)$ or $ac|be \in \mathrm{cl}(Q)$. In the either case, the dyadic inference rule applied to the pair $\{ab|cd, ae|bc\}$ or to $\{ab|cd, ac|be\}$ implies $ae|cd \in \mathrm{cl}(Q)$, and so $\mathrm{cl}(Q)$ satisfies the substitution property. Thus $\mathrm{cl}(Q) = Q(T)$ for a unique tree T. Finally, since $\mathrm{cl}(Q)$ contains a split for each possible quartet, it follows that T must be binary. ■

We now continue with the description of the DCTC algorithm.

Algorithm DCTC.

Step 1. We compute the dyadic closure, cl(Q), of Q.

Step 2.

- **Case 1.** cl(Q) contains a pair of contradictory splits for some quartet: return *Inconsistent*.
- **Case 2.** cl(Q) has no contradictory splits, but fails to have a split for every quartet: Return *Insufficient*.
- **Case 3.** cl(Q) has exactly one split for each quartet: apply standard algorithms [6, 51] to cl(Q) to reconstruct the tree T such that $Q(T) = \text{cl}(Q)$. Return T.

(Case 3 depends upon Lemma 3 above.)

To completely describe the DCTC method we need to specify how we compute the dyadic closure of a set Q of quartet splits.

Efficient computation of dyadic closure. The description we now give of an efficient method for computing the dyadic closure will only actually completely compute the dyadic closure of Q if $\text{cl}(Q) = Q(T)$ for some tree T. Otherwise, cl(Q) will either contain a contradictory pair of splits for some quartet, or cl(Q) will not contain a split for every quartet. In the first of these two cases, the method will return *Inconsistent*, and in the second of these two cases, the method will return *Insufficient*. However, the method can be easily modified to compute cl(Q) for all sets Q.

We will maintain a four-dimensional array `Splits` and constrain $\texttt{Splits}_{\texttt{i,"j,"k,"l}}$ to either be empty, or to contain exactly one split that has been inferred so far for the quartet i, j, k, l. In the event that two conflicting splits are inferred for the same quartet, the algorithm will immediately return *Inconsistent*, and halt. We will also maintain a queue Q_{new} of new splits that must be processed. We initialize `Splits` to contain the splits in the input Q, and we initialize Q_{new} to be Q, ordered arbitrarily.

The dyadic inference rules in equations (8)–(10) show that we infer new splits by combining two splits at a time, where the underlying quartets for the two splits share three leaves. Consequently, each split $ij|kl$ can only be combined with splits on quartets $\{a, i, j, k\}$, $\{a, i, j, l\}$, $\{a, i, k, l\}$, and $\{a, j, k, l\}$, where $a \notin \{i, j, k, l\}$. Consequently, there are only $4(n-4)$ other splits with which any split can be combined using these dyadic rules to generate new splits.

Pop a split $ij|kl$ off the queue Q_{new}, and examine each of the appropriate $4(n-4)$ entries in `Splits`. For each nonempty entry in `Splits` that is examined in this process, compute the $O(1)$ splits that arise from the combination of the two splits. Suppose the combination generates a split $ab|cd$. If $\texttt{Splits}_{a,b,c,d}$ contains a different split from $ab|cd$, then Return *Inconsistent*. If $\texttt{Splits}_{a,b,c,d}$ is empty, then set $\texttt{Splits}_{a,b,c,d} = ab|cd$, and add $ab|cd$ to the queue Q_{new}. Otherwise $\texttt{Splits}_{a,b,c,d}$ already contains the split $ab|cd$, and we do not modify the data structures.

Continue until the queue Q_{new} is empty, or Inconsistency has been observed. If the Q_{new} empties before Inconsistency is observed, then check if every entry of `Splits` is nonempty. If so, then $\text{cl}(Q) = Q(T)$ for some tree; Return `Splits`. If some entry in `Splits` is empty, then return *Insufficient*.

Theorem 5. *The efficient computation of the dyadic closure uses* $O(n^5)$ *time, and at the termination of the algorithm the* `Splits` *matrix is either identically equal to* $\text{cl}(Q)$, *or the algorithm has returned* Inconsistent. *Furthermore, if the algorithm returns* Inconsistent, *then* $\text{cl}(Q)$ *contains a pair of contradictory splits.*

Proof. It is clear that if the algorithm only computes splits using dyadic closure, so that at any point in the application of the algorithm, $\texttt{Splits} \subseteq \text{cl}(Q)$. Consequently, if the algorithm returns *Inconsistent*, then $\text{cl}(Q)$ does contain a pair of contradictory splits. If the algorithm does not return *Inconsistent*, then it is clear from the design that every split which could be inferred using these dyadic rules would be in the `Splits` matrix when the algorithm terminates.

The running time analysis is easy. Every combination of quartet splits takes $O(1)$ time to process. Processing a quartet split involves examining $4(n-4)$ entries in the `Splits` matrix, and hence costs $O(n)$. If a split $ij|kl$ is generated by the combination of two splits, then it is only added to the queue if $\texttt{Splits}_{i,j,k,l}$ is empty when $ij|kl$ is generated. Consequently, at most $O(n^4)$ splits ever enter the queue. ■

We now prove our main theorem of this section:

Theorem 6. *Let* Q *be a set of quartet splits.*

1. *If* $\text{DCTC}(Q) = T$, $\text{DCTC}(Q') = T'$, *and* $Q \subseteq Q'$, *then* $T = T'$.
2. *If* $\text{DCTC}(Q) = \text{Inconsistent}$ *and* $Q \subseteq Q'$, *then* $\text{DCTC}(Q') = \text{Inconsistent}$.
3. *If* $\text{DCTC}(Q) = \text{Insufficient}$ *and* $Q' \subseteq Q$, *then* $\text{DCTC}(Q') = \text{Insufficient}$.
4. *If* $R_T \subseteq Q \subseteq Q(T)$, *then* $\text{DCTC}(Q) = T$.

Proof. Assertion (1) follows from the fact that if $\text{DCTC}(Q) = T$, then the dyadic closure phase of the DCTC algorithm computes exactly one split for every quartet, so that $\text{cl}(Q) = Q(T)$ by Lemma 3. Therefore, if $Q \subseteq Q'$, then $\text{cl}(Q) \subseteq \text{cl}(Q')$, so that $Q(T) \subseteq \text{cl}(Q') = Q(T')$. Since T and T' are binary trees, it follows that $Q(T) = Q(T')$ and $T = T'$.

Assertion (2) follows from the fact that if $\text{DCTC}(Q) = \textit{Inconsistent}$, then $\text{cl}(Q)$ contains two contradictory splits for the same quartet. If $Q \subseteq Q'$, then $\text{cl}(Q')$ also contains the same two contradictory splits, and so $\text{DCTC}(Q') = \textit{Inconsistent}$.

Assertion (3) follows from the fact that if $\text{DCTC}(Q) = \textit{Insufficient}$, then $\text{cl}(Q)$ does not contain contradictory pairs of splits, and also lacks a split for at least one quartet. If $Q' \subseteq Q$, then $\text{cl}(Q')$ also does not contain contradictory pairs of splits and also lacks a split for some quartet. Consequently, $\text{DCTC}(Q') = \textit{Insufficient}$.

Assertion (4) follows from Lemma 2 and Assertion (1). ■

Note that $\text{DCTC}(Q) = \textit{Insufficient}$ does not actually imply that $Q \subset Q(T)$ for any tree; that is, it may be that $Q \not\subseteq Q(T)$ for any tree, but $\text{cl}(Q)$ may not contain any contradictory splits!

5. DYADIC CLOSURE METHOD

We now describe a new method for tree reconstruction, which we call the *Dyadic Closure Method*, or DCM.

Suppose T is a fixed binary tree. From the previous section, we know that if we can find a set Q of quartet splits such that $R_T \subseteq Q \subseteq Q(T)$, then DCTC($Q$) will reconstruct T.

One approach to find such a set Q would be to let Q be the set of splits (computed using the Four-Point Method) on all possible quartets. However, it is possible that the sequence length needed to ensure that *every* quartet is accurately analyzed might be too large to obtain accurate reconstructions of large trees, or of trees containing short edges.

The approach we take in the Dyadic Closure Method is to use sets of quartet splits based upon the quartets whose topologies should be easy to infer from short sequences, rather than upon all possible quartets. (By contrast, other quartet based methods, such as Quartet Puzzling [47, 48], the Buneman tree construction [7], etc. infer quartet splits for all the possible quartets in the tree.) Basing the tree reconstruction upon properly selected sets of quartets makes it possible to expect, even from short sequences, that all the quartet splits inferred for the selected subset of quartets will be valid.

Since what we need is a set Q such that $R_T \subseteq Q \subseteq Q(T)$, we need to ensure that we pick a *large enough* set of quartets so that it contains all of R_T, and yet not too large that it contains any invalid quartet splits. Surprisingly, obtaining such a set Q is quite easy (once the sequences are long enough), and we describe a greedy approach which accomplishes this task. We will also show that the greedy approach can be implemented very efficiently, so that not too many calls to the DCTC algorithm need to be made in order to reconstruct the tree, and analyze the sequence length needed for the greedy approach to succeed with $1 - o(1)$ probability.

We now describe how this is accomplished.

Definition 2. [Q_w, and the *width* of a quartet]. The *width* of a quartet i, j, k, l is defined to be the maximum of $h^{ij}, h^{ik}, h^{il}, h^{jk}, h^{jl}, h^{kl}$, where h^{ij} denotes the dissimilarity score between sequences i and j (see Section 2). For each quartet whose width is at most w, compute all feasible splits on that quartet using the four-point method. Q_w is defined to be the set of all such reconstructed splits.

(We note that we could also compute the split for a given quartet of sequences in any number of ways, including maximum likelihood estimation, parsimony, etc., but we will not explore these options in this paper.)

For large enough values of w, Q_w will with high probability contain invalid quartet splits (unless the sequences are very long), while for very small values of w, Q_w will with high probability only contain valid quartet splits (unless the sequences are very short). Since our objective is a set of quartet splits Q such that $R_T \subseteq Q \subset Q(T)$, what we need is a set Q_w such that Q_w contains only valid quartet splits, and yet w is large enough so that all representative quartets are contained in Q_w as well.

We define sets

$$\mathscr{A} = \{w \in \{h^{ij}: 1 \le i, j \le n\}: R_T \subseteq Q_w\}, \tag{12}$$

and

$$\mathscr{B} = \{w \in \{h^{ij}: 1 \le i, j \le n\}: Q_w \subseteq Q(T)\}. \tag{13}$$

In other words, $\mathscr{A}$ is the set of widths w (drawn from the set of dissimilarity scores) which equal to exceed the largest width of any representative quartet, and $\mathscr{B}$ is the set of widths (drawn from the same set) such that all quartet splits of that dissimilarity score are correctly analyzed by the Four-Point Method.

It is clear that $\mathscr{B}$ is an initial segment in the list of widths, and that $\mathscr{A}$ is a final segment (these segments can be empty). It is easy to see that if $w \in \mathscr{A} \cap \mathscr{B}$, then $\mathrm{DCTC}(Q_w) = T$. Thus, if the sequences are long enough, we can apply DCTC to each of the $O(n^2)$ sets Q_w of splits, and hence reconstruct the tree properly. However, the sequences may not be long enough to ensure that such a w exists; i.e., $\mathscr{A} \cap \mathscr{B} = \emptyset$ is possible! Consequently, we will require that $\mathscr{A} \cap \mathscr{B} \neq \emptyset$, and state this requirement as an hypothesis (later, we will show in Theorem 9 that this hypothesis holds with high probability for sufficiently long sequences),

$$\mathscr{A} \cap \mathscr{B} \neq \emptyset. \tag{14}$$

When this hypothesis holds, we clearly have a polynomial time algorithm, but we can also show that the DCTC algorithm enables a binary search approach over the realized widths values, so that instead of $O(n^2)$ calls to the DCTC algorithm, we will have only $O(\log n)$ such calls.

Recall that $\mathrm{DCTC}(Q_w)$ is either a tree T, Inconsistent, or Insufficient.

- Insufficient. This indicates that w is too small, because not all representative quartet splits are present, and we should increase w.
- Tree output. If this happens, the quartets are consistent with a unique tree, and that tree is returned.
- Inconsistent. This indicates that the quartet splits are incompatible, so that no tree exists which is consistent with each of the constraints. In this case, we have computed the split of at least one quartet incorrectly. This indicates that w is too large, and we should decrease w.

If not all representative quartets are inferred correctly, then every set Q_w will be either insufficient or inconsistent with T, perhaps consistent with a different tree. In this case the sequences are too short for the DCM to reconstruct a tree accurately.

We summarize our discussion as follows:

Dyadic Closure Method.

Step 1. Compute the distance matrices d and h (recall that d is the matrix of corrected empirical distances, and h is the matrix of normalized Hamming distances, i.e., the *dissimilarity* score).

Step 2. Do a *binary search* as follows: for $w \in \{h^{ij}\}$, determine Q_w. If $\mathrm{DCTC}(Q_w) = T$, for some tree T, then Return T. If DCTC returns *Inconsistent*, then w is too large; decrease w. If DCTC returns *Insufficient*, then w is too small; increase w.

Step 3. If for all w, DCTC applied to Q_w returns *Insufficient* or *Inconsistent*, then Return *Fail*.

We now show that this method accurately reconstructs the tree T if $\mathscr{A} \cap \mathscr{B} \neq \emptyset$ [i.e., if hypothesis (14) holds].

Theorem 7. *Let T be a fixed binary tree. The Dyadic Closure Method returns T if hypothesis* (14) *holds, and runs in $O(n^5 \log n)$ time on any input.*

Proof. If $w \in \mathscr{A} \cap \mathscr{B}$, then DCTC applied to Q_w returns the correct tree T by Theorem 6. Hypothesis (14) implies that $\mathscr{A} \cap \mathscr{B} \neq \emptyset$, hence the Dyadic Closure Method returns a tree if it examines any width in that intersection; hence, we need only prove that DCM either examines a width in that intersection, or else reconstructs the correct tree for some other width. This follows directly from Theorem 6.

The running time analysis is easy. Since we do a binary search, the DCTC algorithm is called at most $O(\log n)$ times. The dyadic closure phase of the DCTC algorithm costs $O(n^5)$ time, by Lemma 5, and reconstructing the tree T from $\mathrm{cl}(Q)$ uses at most $O(n^5)$ time using standard techniques. ■

Note that we have only guaranteed performance for DCM when $\mathscr{A} \cap \mathscr{B} \neq \emptyset$; indeed, when $\mathscr{A} \cap \mathscr{B} = \emptyset$, we have no guarantee that DCM will return the correct tree. In the following section, we discuss the ramifications of this requirement for accuracy, and show that the sequence length needed to guarantee that $\mathscr{A} \cap \mathscr{B} \neq \emptyset$ with high probability is actually not very large.

6. PERFORMANCE OF DYADIC CLOSURE METHOD FOR TREE RECONSTRUCTION UNDER THE NEYMAN 2-STATE MODEL

In this section we analyze the performance of a distance-based application of DCM to reconstruct trees under the Neyman 2-state model under two standard distributions.

6.1. Analysis of the Dyadic Closure Method

Our analysis of the Dyadic Closure Method has two parts. In the first part, we establish the probability that the estimation (using the Four-Point Method) of the split induced by a given quartet is correct. In the second part, we establish the probability that the greedy method we use contains all short quartets but no incorrectly analyzed quartet.

Our analysis of the performance of the DCM method depends heavily on the following two lemmas:

Lemma 4 [Azuma–Hoeffding inequality, see [3]]. *Suppose $X = (X_1, X_2, \ldots, X_k)$ are independent random variables taking values in any set S, and L: $S^k \to \mathbb{R}$ is any function that satisfies the condition*: $|L(\mathbf{u}) - L(\mathbf{v})| \leq t$ *whenever* $\mathbf{u}$ *and* $\mathbf{v}$ *differ at just*

one coordinate. Then,

$$\mathbb{P}\big[L(\mathbf{X}) - \mathbb{E}\big[L(\mathbf{X})\big] \geq \lambda\big] \leq \exp\left(-\frac{\lambda^2}{2t^2k}\right),$$

$$\mathbb{P}\big[L(\mathbf{X}) - \mathbb{E}\big[L(\mathbf{X})\big] \leq -\lambda\big] \leq \exp\left(-\frac{\lambda^2}{2t^2k}\right). \quad \blacksquare$$

We define the (standard) L_∞ metric on distance matrices, $L_\infty(d, d') = \max_{ij}|d_{ij} - d'_{ij}|$. The following discussion relies upon definitions and notations from Section 2.

Lemma 5. *Let T be an edge weighted binary tree with four leaves i, j, k, l, let D be the additive distance matrix on these four leaves defined by T, and let x be the weight on the single internal edge in T. Let d be an arbitrary distance matrix on the four leaves. Then the Four-Point Method infers the split induced by T from d if $L_\infty(d, D) < x/2$.*

Proof. Suppose that $L_\infty(d, D) < x/2$, and assume that T has the valid split $ij|kl$. Note that the four-point method will return a single quartet, split $ij|kl$ if and only if $d_{ij} + d_{kl} < \min\{d_{ik} + d_{jl}, d_{il} + d_{jk}\}$. Note that since $ij|kl$ is a valid quartet split in T, $D_{ij} + D_{kl} + 2x = D_{ik} + D_{jl} = D_{il} + D_{jk}$. Since $L_\infty(d, D) < x/2$, it follows that

$$d_{ij} + d_{kl} < D_{ij} + D_{kl} + x,$$
$$d_{ik} + d_{jl} > D_{ik} + D_{jl} - x,$$

and

$$d_{il} + d_{jk} > D_{il} + D_{jk} - x,$$

with the consequence that $d_{ij} + d_{kl}$ is the (unique) smallest of the three pairwise sums. $\blacksquare$

Recall that DCM applied to the Neyman 2-state model computes quartet splits using the four-point method (FPM).

Theorem 8. *Assume that z is a lower bound for the transition probability of any edge of a tree T in the Neyman 2-state model, $y \geq \max E^{ij}$ is an upper bound on the compound changing probability over all ij paths in a quartet q of T. The probability that FPM fails to return the correct quartet split on q from k sites is at most*

$$18\exp\frac{-\left(1-\sqrt{1-2z}\right)^2(1-2y)^2k}{8}. \tag{15}$$

Proof. First observe from formula (1) that z is also a lower bound for the compound changing probability for the path connecting *any* two vertices of T. We know that FPM returns the appropriate subtree given the additive distances D_{ij}; furthermore, if $|d_{ij} - D_{ij}| \leq -\frac{1}{4}\log(1-2z)$ for all i, j, then FPM also returns the

appropriate subtree on all $ijkl$, by Lemma 5. Consequently,

$$\mathbb{P}[\text{FPM errs}] \le \mathbb{P}\left[\exists i, j: |D_{ij} - d_{ij}| > -\tfrac{1}{4}\log(1-2z)\right]. \tag{16}$$

Hence by (16), we have

$$\mathbb{P}[\text{FPM errs}] \le \sum_{ij} \mathbb{P}\left[|D_{ij} - d_{ij}| > -\tfrac{1}{4}\log(1-2z)\right]. \tag{17}$$

For convenience, we drop the subscripts when we analyze the events in (17) and just write D and d; we write p for the corresponding transition probability E^{ij} and $\hat{p}$ for the relative frequency h^{ij}. By simple algebra,

$$|D-d| = \frac{1}{2}\log\frac{1-2p}{1-2\hat{p}}, \quad \text{if } p < \hat{p}, \tag{18}$$

$$|D-d| = \frac{1}{2}\log\frac{1-2\hat{p}}{1-2p}, \quad \text{if } p \ge \hat{p}. \tag{19}$$

Now we consider the probability that the Four-Point Method fails, i.e., the event estimated in (17). If $p \ge \hat{p}$, then formula (19) applies, so that $\mathbb{P}[\text{FPM errs}]$ is algebraically equivalent to

$$p - \hat{p} \ge \tfrac{1}{2}\left[(1-2z)^{-1/2} - 1\right](1-2p). \tag{20}$$

This can then be analyzed using Lemma 4. The other case is where $p < \hat{p}$. In this case, formula (18) applies, and $\mathbb{P}[\text{FPM errs}]$ is algebraically equivalent to

$$\frac{\hat{p}-p}{1-2\hat{p}} \ge \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right]. \tag{21}$$

Select an arbitrary positive number ϵ. Then $\hat{p} - p \ge (1-2p)\epsilon$ with probability

$$\exp\frac{-\epsilon^2(1-2p)^2 k}{2}, \tag{22}$$

by Lemma 4. If $\hat{p} - p < (1-2p)\epsilon$, then

$$\frac{1}{1-2\hat{p}} < \frac{1}{(1-2p) - 2\epsilon(1-2p)} = \frac{1}{(1-2p)}\,\frac{1}{(1-2\epsilon)}.$$

Hence

$$\begin{aligned}
&\mathbb{P}\left[\frac{\hat{p}-p}{1-2\hat{p}} \ge \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right]\right] \\
&\quad \le \mathbb{P}\left[\frac{\hat{p}-p}{(1-2p)(1-2\epsilon)} \ge \frac{1}{2}\left[(1-2z)^{-1/2} - 1\right]\right] + \exp\frac{-\epsilon^2(1-2p)^2 k}{2} \\
&\quad \le \exp\frac{-\epsilon^2(1-2p)^2 k}{2} \qquad (23) \\
&\qquad + \exp\frac{-(1-2p)^2(1-2\epsilon)^2\left[(1-2z)^{-1/2} - 1\right]^2 k}{8}. \qquad (24)
\end{aligned}$$

Note that $\epsilon = (\frac{1}{2})[1-(1-2z)^{1/2}]$ is the optimal choice. Formulae (22–24) contribute each the same exponential expression to the error, and (16) or (17) multiplies it by 6, due to the six pairs in the summation. ■

This allows us to state our main result. First, recall the definition of *depth* from Section 2.

Theorem 9. *Suppose k sites evolve under the Neyman 2-state model on a binary tree T, so that for all edges e, $p(e) \in [f, g]$, where we allow f, g to be functions of n. Then the dyadic closure method reconstructs T with probability $1 - o(1)$, if*

$$k > \frac{c \cdot \log n}{\left(1-\sqrt{1-2f}\right)^2 (1-2g)^{4\,\mathrm{depth}(T)+6}}, \tag{25}$$

where c is a fixed constant.

Proof. It suffices to show that hypothesis (14) holds. For k evolving sites (i.e., sequences of length k), and $\tau > 0$, let us define the following two sets, $S_\tau = \{\{i, j\}: h^{ij} < 0.5 - \tau\}$ and

$$Z_\tau = \left\{ q \in \binom{[n]}{4} : \text{for all } i, j \in q, \{i, j\} \in S_{2\tau} \right\},$$

and the following four events,

$$A = Q_{\mathrm{short}}(T) \subseteq Z_\tau, \tag{26}$$

$$B_q = \text{FPM correctly returns the split of the quartet } q \in \binom{[n]}{4}, \tag{27}$$

$$B = \bigcap_{q \in Z_\tau} B_q, \tag{28}$$

$$C = S_{2\tau} \text{ contains all pairs } \{i, j\} \text{ with } E^{ij} < 0.5 - 3\tau \text{ and no pair } \{i, j\} \text{ with } E^{ij} \geq 0.5 - \tau. \tag{29}$$

Thus, $\mathbb{P}[\mathscr{A} \cap \mathscr{B} \neq \varnothing] \geq \mathbb{P}[A \cap B]$. Define

$$\lambda = (1-2g)^{2\,\mathrm{depth}(T)+3}. \tag{30}$$

We claim that

$$\mathbb{P}[C] \geq 1 - (n^2 - n) e^{-\tau^2 k/2}, \tag{31}$$

and

$$\mathbb{P}[A|C] = 1, \quad \text{if } \tau \leq \frac{\lambda}{6}. \tag{32}$$

To establish (31), first note that h^{ij} satisfies the hypothesis of the Azuma–Hoeffding inequality (Lemma 4 with X_i the sequence of states for site i and $t = 1/k$).

Suppose $E^{ij} \geq .5 - \tau$. Then,

$$\mathbb{P}\big[\{i,j\} \in S_{2\tau}\big] = \mathbb{P}\big[h^{ij} < 0.5 - 2\tau\big]$$
$$\leq \mathbb{P}\big[h^{ij} - E^{ij} \leq 0.5 - 2\tau - E^{ij}\big] \leq \mathbb{P}\big[h^{ij} - \mathbb{E}\big[h^{ij}\big] \leq -\tau\big] \leq e^{-\tau^2 k/2}.$$

Since there are at most $\binom{n}{2}$ pairs $\{i,j\}$, the probability that at least one pair $\{i,j\}$ with $E^{ij} \geq 0.5 - \tau$ lies in $S_{2\tau}$ is at most $\binom{n}{2} e^{-\tau^2 k/2}$. By a similar argument, the probability that $S_{2\tau}$ fails to contain a pair $\{i,j\}$ with $E^{ij} < 0.5 - 3\tau$ is also at most $\binom{n}{2} e^{-\tau^2 k/2}$. These two bounds establish (31).

We now establish (32). For $q \in R(T)$ and $i,j \in q$, if a path $e_1 e_2 \cdots e_t$ joins leaves i and j, then $t \leq 2\,\mathrm{depth}(T) + 3$ by the definition of $R(T)$. Using these facts, (1), and the bound $p(e) \leq g$, we obtain $E^{ij} = 0.5[1 - (1-2p_1)\cdots(1-2p_t)] \leq 0.5(1-\lambda)$. Consequently, $E^{ij} < 0.5 - 3\tau$ (by assumption that $\tau \leq \lambda/6$) and so $\{i,j\} \in S_{2\tau}$ once we condition on the occurrence of event C. This holds for all $i,j \in q$, so by definition of Z_τ we have $q \in Z_\tau$. This establishes (32).

Define a set,

$$X = \left\{ q \in \binom{[n]}{4} : \max\{E^{ij} : i,j \in q\} < 0.5 - \tau \right\},$$

(note that X is not a random variable, while Z_τ, S_τ are). Now, for $q \in X$, the induced subtree in T has mutation probability at least $f(n)$ on its central edge, and mutation probability of no more than $\max\{E^{ij} : i,j \in q\} < 0.5 - \tau$ on any pendant edge. Then, by Theorem 8 we have

$$\mathbb{P}\big[B_q\big] \geq 1 - 18\exp\frac{-\big(1-\sqrt{1-2f}\big)^2 \tau^2 k}{8}. \tag{33}$$

whenever $q \in X$. Also, the occurrence of event C implies that

$$Z_\tau \subseteq X, \tag{34}$$

since if $q \in Z_\tau$, and $i,j \in q$, then $i,j \in S_{2\tau}$, and then (by event C), $E^{ij} < 0.5 - \tau$, hence $q \in X$. Thus, since $B = \bigcap_{q \in Z_\tau} B_q$, we have

$$\mathbb{P}[B \cap C] = \mathbb{P}\left[\left(\bigcap_{q \in Z_\tau} B_q\right) \cap C\right] \geq \mathbb{P}\left[\left(\bigcap_{q \in X} B_q\right) \cap C\right],$$

where the second inequality follows from (34), as this shows that when C occurs, $\bigcap_{q \in Z_\tau} B_q \supseteq \bigcap_{q \in X} B_q$. Invoking the Bonferonni inequality, we deduce that

$$\mathbb{P}[B \cap C] \geq 1 - \sum_{q \in X} \mathbb{P}\big[\overline{B_q}\big] - \mathbb{P}[\overline{C}]. \tag{35}$$

Thus, from above,

$$\mathbb{P}[A \cap B] \geq \mathbb{P}[A \cap B \cap C] = P[B \cap C],$$

(since $\mathbb{P}[A|C] = 1$), and so, by (33) and (35),

$$\mathbb{P}[A \cap B] \geq 1 - 18\binom{n}{4}\exp\frac{-\left(1-\sqrt{1-2f}\right)^2\tau^2 k}{8} - (n^2-n)e^{-\tau^2 k/2}.$$

Formula (25) follows by an easy calculation. ■

6.2. Distributions on Trees

In the previous section we provided an upper bound on the sequence length that suffices for the Dyadic Closure Method to achieve an accurate estimation with high probability, and this upper bound depends critically upon the *depth* of the tree. In this section, we determine the depth of a random tree under two simple models of random binary trees.

These models are the *uniform* model, in which each tree has the same probability, and the *Yule–Harding* model, studied in [2, 8, 27] (the definition of this model is given later in this section). This distribution is based upon a simple model of speciation, and results in "bushier" trees than the uniform model. The following results are needed to analyze the performance of our method on random binary trees.

Theorem 10.

- **(i)** *For a random semilabeled binary tree T with n leaves under the uniform model,* $\text{depth}(T) \leq (2 + o(1))\log_2 \log_2(2n)$ *with probability* $1 - o(1)$.
- **(ii)** *For a random semilabeled binary tree T with n leaves under the Yule–Harding distribution, after suppressing the root,* $\text{depth}(T) = (1 + o(1))\log_2 \log_2 n$ *with probability* $1 - o(1)$.

Proof. This proof relies upon the definition of an *edi-subtree*, which we now define. If (a, b) is an edge of a tree T, and we delete the edge (a, b) but not the endpoints a or b, then we create two subtrees, one containing the node a and one containing the node b. By rooting each of these subtrees at a (or b), we obtain an edge-deletion induced subtree, or "edi-subtree."

We now establish (i). Recall that the number of all semilabeled binary trees is $(2n-5)!!$ Now there is a unique (unlabeled) binary tree F on $2^t + 1$ leaves with the following description: one endpoint of an edge is identified with the degree 2 root of a complete binary tree with 2^t leaves. The number of semilabeled binary trees whose underlying topology is F is $(2^t+1)!/2^{2^t-1}$. This is fairly easy to check and this also follows from Burnside's lemma as applied to the action of the symmetric group on trees, as was first observed by [32] in this context. A rooted semilabeled binary forest is a forest on n labeled leaves, m trees, such that every tree is either a single leaf or a binary tree which is rooted at a vertex of degree 2. It was proved by Carter et al. [11] that the number of rooted semilabeled binary forests is

$$N(n, m) = \binom{2n-m-1}{m-1}(2n-2m-1)!!.$$

Now we apply the probabilistic method. We want to set a number t large enough, such that the total number of edi-subtrees of depth at least t in the set of all semilabeled binary trees on n vertices is $o((2n-5)!!)$. The theorem then follows for this number t. We show that some $t=(2+o(1))\log_2\log_2(2n)$ suffices. We count ordered pairs in two ways, as usual: Let E_t denote the number of edi-subtrees of depth at least t (edi-subtrees induced by internal edges and leaf edges combined) counted over of all semilabeled trees. Then E_t is equal to the number of ways to construct a rooted semilabeled binary forest on n leaves of 2^t+1 trees, then use the 2^t+1 trees as leaf set to create all F-shaped semilabeled trees (as described above), with finally attaching the leaves of F to the roots of the elements of the forest. Then $E_t=((2^t+1)!/2^{2^t-1})N(n,2^t+1)$. Hence everything boils down to finding a t for which

$$\frac{(2^t+1)!}{2^{2^t-1}}\binom{2n-2^t-2}{2^t}(2n-2^{t+1}-3)!!=o\big((2n-5)!!\big).$$

Clearly $t=(2+\delta)\log_2\log_2(2n)$ suffices.

We now consider (ii). First we describe the proof for the usual rooted Yule–Harding trees. These trees are defined by the following construction procedure. Make a random permutation $\pi_1,\pi_2,\ldots,\pi_n$ of the n leaves, and join π_1 and π_2 by edges t a root R of degree 2. Add each of the remaining leaves sequentially, by randomly (with the uniform probability) selecting an edge incident to a leaf in the tree already constructed, subdividing the edge, and make π_i adjacent to the newly introduced node. For the depth of a Yule–Harding tree, consider the following recursive labeling of the edges of the tree. Call the edge $\pi_i R$ (for $i=1,2$) "i new." When π_i is added ($i\geq 3$) by insertion into an edge with label "j new," we given label "i new" to the leaf edge added, give label "j new" to the leaf part of the subdivided edge, and turn the label "j new" into "j old" on the other part of the subdivided edge. Clearly, after l insertions, all numbers $1,2,\ldots,l$ occur exactly once with label new, in each occasion labeling leaf edges. The following which may help in understanding the labeling: edges with "old" label are exactly the internal edges and j is the smallest label in the subtree separated by an edge labeled "j old" from the root R, any time during the labeling procedure.

We now derive an upper bound for the probability that an edi-subtree of depth d develops. If it happens, then a leaf edge inserted at some point has to grow a deep edi-subtree on one side. Let us denote by T_i^R the rooted random tree that we already obtained with i leaves. Consider the probability that the most recently inserted edge i new ever defines an edi-subtree with depth d. Such an event can happen in two ways: this edi-subtree may emerge on the leaf side of the edge or on the tree side of the edge (these sides are defined when the edge is created). Let us denote these probabilities by $\mathbb{P}[i,\mathrm{OUT}|T_i^R]$ and $\mathbb{P}[i,\mathrm{IN}|T_i^R]$, since these probabilities may depend on the shape of the tree already obtained (and, in fact, the second probability does so depend on the shape of T_i^R). We estimate these quantities with tree-independent quantities.

For the moment, take for granted the following inequalities,

$$\mathbb{P}\big[i,\mathrm{OUT}|T_i^R\big]\leq\mathbb{P}\big[i,\mathrm{IN}|T_i^R\big], \tag{36}$$

$$\mathbb{P}\big[i,\mathrm{IN}|T_i^R\big]\leq\epsilon(d,n), \tag{37}$$

for some function $\epsilon(d,n)$ defined below. Clearly,

$$\mathbb{P}[\exists \text{ depth } d \text{ edi-subtree}] \le \sum_{i=1}^{n} \sum_{T_i^R} \mathbb{P}\left[i, \text{OUT} | T_i^R\right] \mathbb{P}\left[T_i^R\right] + \mathbb{P}\left[i, \text{IN} | T_i^R\right] \mathbb{P}\left[T_i^R\right], \tag{38}$$

and using (36) and (37), (38) simplifies to

$$\mathbb{P}[\exists \text{ depth } d \text{ edi-subtree}] \le 2n\epsilon(d,n). \tag{39}$$

We now find an appropriate $\epsilon(d,n)$.

For convenience we assume that $2^s = n-2$, since it simplifies the calculations. Set $k = 2^{d-1}-1$, it is clear that at least k properly placed insertions are needed to make the current edge "i new" have depth d on its tree side. Indeed, π_i was inserted into a leaf edge labeled "j new" and one side of this leaf edge is still a leaf, which has to develop into depth $d-1$, and this development requires at least k new leaf insertions.

Focus now entirely on the k insertions that change "j new" into an edi-subtree of depth $d-1$. Rank these insertions by $1,2,\ldots,k$ in order, and denote by 0 the original "j new" leaf edge. Then any insertion ranked $i \ge 1$ may go into one of those ranked $0,1,\ldots,i-1$. Call the function which tells for $i = 1,2,\ldots,k$, which depth i is inserted into, a *core*. Clearly, the number of cores is at most k^k.

We now estimate the probability that a fixed core emerges. For any fixed $i_1 < i_2 < \cdots < i_k$, the probability that inserting π_{i_j} will make the insertion enumerated under depth j, for all $j = 1,2,\ldots,k$, is at most

$$\frac{1}{i_1 - 1} \cdot \frac{1}{i_2 - 1} \cdots \frac{1}{i_k - 1},$$

by independence. Summarizing our observations,

$$\begin{aligned} \mathbb{P}\left[i, \text{IN} | T_i^R\right] &\le k^k \sigma_{n-i}^k \left(\frac{1}{i}, \frac{1}{i+1}, \ldots, \frac{1}{n-1}\right) \\ &\le k^k \sigma_{n-2}^k \left(\frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{n-1}\right), \end{aligned} \tag{40}$$

where σ_m^k is the symmetric polynomial of m variables of degree k. We set $\epsilon(n,d) = \sigma_{n-2}^k(\frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{n-1})$. To estimate (40), observe that any term in $\sigma_{n-2}^k(\frac{1}{2}, \frac{1}{3}, \ldots, \frac{1}{n-1})$ can be described as having exactly a_i reciprocals of integers substituted from the interval $(2^{-(i+1)}, 2^{-i}]$. The point is that those reciprocals differ little in each of those intervals, and hence a close estimate is possible. A generic term of σ_{n-2}^k as described above is estimated from above by

$$2^{-(1\cdot a_1 + 2\cdot a_2 + \cdots + (s-1)a_{s-1})}. \tag{41}$$

Hence $\epsilon(n,d)$ is at most

$$\sum_{\substack{a_1+a_2+\cdots+a_{s-1}=k \\ a_i\le 2^i}} \binom{2}{a_1}\binom{4}{a_2}\binom{8}{a_3}\cdots\binom{2^{s-1}}{a_{s-1}} 2^{-(1\cdot a_1+2\cdot a_2+\cdots+(s-1)a_{s-1})}, \quad (42)$$

by (41). Since

$$\binom{2^i}{a_i}2^{-ia_i} \le \frac{1}{a_i!},$$

(42) is less than or equal

$$\sum_{\substack{a_1+a_2+\cdots+a_{s-1}=k \\ a_i\le 2^i}} \frac{1}{a_1!a_2!\cdots a_{s-1}!}. \quad (43)$$

Observe that the number of terms in (43) is at most the number of compositions of k into $s-1$ terms,

$$\binom{k+s-2}{s-2}.$$

The product of factorials is minimized (irrespective of $a_i \le 2^i$) if all a_is are taken equal. Hence, setting $k = s^{1+\delta}$ for any fixed $\delta > 0$, (43) is at most

$$\left(\frac{(k+s-2)^{s-2}}{(s-2)!}\right) \Big/ \left(\left(\frac{k}{s-1}\right)!^k\right),$$

and hence

$$\epsilon(n,d) \le k^k \left(\frac{(k+s-2)^{s-2}}{(s-2)!}\right) \Big/ \left(\left(\frac{k}{s-1}\right)!^k\right) \le n^{-c\log n \log\log n},$$

and (39) goes to zero. For the depth d, our calculation yields $(1+\delta+o(1))\log_2\log_2 n$ with probability $1-o(1)$.

We leave the establishment of (36) to the reader. Now, to obtain a similar result for unrooted Yule–Harding trees, just repeat the argument above, but use the unrooted T_i instead of the rooted T_i^R. The probability of any T_i is the sum of probabilities of $2i-3$ rooted T_i^Rs, since the root could have been on every edge of T_i. Hence formula (37) has to be changed for $\mathbb{P}[i, \mathrm{IN}|T_i] \le (2n-3)\epsilon(d,n)$. With this change the same proof goes through, and the threshold does not change. ■

6.3. The Performance of Dyadic Closure Method and Two Other Distance Methods for Inferring Trees in the Neyman 2-State Model

In this section we describe the convergence rate for the DCM method, and compare it briefly to the rates for two other distance-based methods, the Agarwala et al. 3-approximation algorithm [1] for the L_∞ nearest tree, and neighbor-joining

[40]. We make the natural assumption that all methods use the same corrected empirical distances from Neyman 2-state model trees.

The neighbor-joining method is perhaps the most popular distance-based method used in phylogenetic reconstruction, and in many simulation studies (see [33, 34, 41] for an entry into this literature) it seems to outperform other popular distance based methods. The Agarwala et al. algorithm [1] is a distance-based method which provides a 3-approximation to the L_∞ nearest tree problem, so that it is one of the few methods which provide a provable performance guarantee with respect to any relevant optimization criterion. Thus, these two methods are two of the most promising distance-based methods against which to compare our method. Both these methods use polynomial time.

In [23], Farach and Kannan analyzed the performance of the 3-approximation algorithm with respect to tree reconstruction in the Neyman 2-state model, and proved that the Agarwala et al. algorithm converged quickly for the *variational distance* (a related but different concern). Recently, Kannan [35] extended the analysis and obtained the following counterpart to (25): If T is a Neyman 2-state model tree with mutation rates in the range $[f, g]$, and if sequences of length k' are generated on this tree, where

$$k' > \frac{c' \cdot \log n}{f^2(1-2g)^{2\,\mathrm{diam}(T)}}, \tag{44}$$

for an appropriate constant c', and were $\mathrm{diam}(T)$ denotes the "diameter" of T, then with probability $1 - o(1)$ the result of applying Agarwala et al. to corrected distances will be a tree with the same topology as the model tree. In [5], Atteson proved an identical statement for neighbor-joining, though with a different constant (the proved constant for neighbor-joining is smaller than the proved constant for the Agarwala et al. algorithm).

Comparing this formula to (25), we note that the comparison of depth and diameter is the issue, since $(1 - \sqrt{1-2f})^2 = \Theta(f^2)$ for small f. It is easy to see that $\mathrm{diam}(T) \geq 2\,\mathrm{depth}(T)$ for binary trees T, but the diameter of a tree can in fact be quite large (up to $n-1$), while the depth is never more than $\log n$. Thus, for every fixed range of mutation probabilities, the sequence length that suffices to guarantee accuracy for the neighbor-joining or Agarwala et al. algorithms can be quite large (i.e., it can grow exponentially in the number of leaves), while the sequence length that suffices for the Dyadic Closure Method will never grow more than polynomially. See also [20, 21, 39] for further studies on the sequence length requirements of these methods.

The following table summarizes the worst case analysis of the sequence length that suffices for the dyadic closure method to obtain an accurate estimation of the tree, for a fixed and a variable range of mutation probabilities. We express these sequence lengths as functions of the number n of leaves, and use results from (25) and Section 6.2 on the depth of random binary trees. "Best case" (respectively, "worst case") trees refers to best case (respectively worst case) *shape* with respect to the sequence length needed to recover the tree as a function of the number n of leaves. Best case trees for DCM are those whose depth is small with respect to the number of leaves; these are the *caterpillar* trees, i.e., trees which are formed by

TABLE 1 Sequence Length Needed by Dyadic Closure Method to Return Trees under the Neyman 2-State Model

	Range of Mutation Probabilities on Edges:	
	$[f, g]$ f, g are constants	$\left[\frac{1}{\log n}, \frac{\log\log n}{\log n}\right]$
Worst case trees	polynomial	polylog
Best case trees	logarithmic	polylog
Random (uniform) trees	polylog	polylog
Random (Yule–Harding) trees	polylog	polylog

attaching n leaves to a long path. Worst case trees for DCM are those trees whose depth is large with respect to the number of leaves; these are the *complete binary trees*. All trees are assumed to be binary.

One has to keep in mind that comparison of performance guarantees for algorithms do not substitute for comparison of performances. Unfortunately, no analysis is available yet on the performance of the Agarwala et al. and neighbor-joining algorithms on random trees, therefore we had to use their worst case estimates also for the case of random leaves.

7. SUMMARY

We have provided upper and lower bounds on the sequence length k for accurate tree reconstruction, and have shown that in certain cases these two bounds are surprisingly close in their order of growth with n. It is quite possible that even better upper bounds could be obtained by a tighter analysis of our DCM approach, or perhaps by analyzing other methods.

Our results may provide a nice analytical explanation for some of the surprising results of recent simulation studies (see, for example, [30]) which found that trees on hundreds of species could be accurately reconstructed from sequences of only a few thousand sites long. For molecular biology the results of this paper may be viewed, optimistically, as suggesting that large trees can be reconstructed accurately from realistic length sequences. Nevertheless, some caution is required, since the evolution of real sequences will only be approximately described by these models, and the presence of very short and/or very long edges will call for longer sequence lengths.

ACKNOWLEDGMENTS

Thanks are due to Sampath Kannan for extending the analysis of [23] to consider the topology estimation, and to David Bryant and Éva Czabarka for proofreading the manuscript.

Tandy Warnow was supported by an NSF Young Investigator Award CCR-9457800, a David and Lucille Packard Foundation fellowship, and generous research support from the Penn Research Foundation and Paul Angello. Michael Steel was supported by the New Zealand Marsden Fund and the New Zealand Ministry of Research, Science and Technology. Péter L. Erdős was supported in part by the Hungarian National Science Fund contracts T 016 358. László Székely was supported by the National Science Foundation grant DMS 9701211, the Hungarian National Science Fund contracts T 016 358 and T 019 367, and European Communities (Cooperation in Science and Technology with Central and Eastern European Countries) contract ERBCIPACT 930 113. This research started in 1995 when the authors enjoyed the hospitality of DIMACS during the Special Year for Mathematical Support to Molecular Biology, and was completed in 1997 while enjoying the hospitality of Andreas Dress, at Universität Bielefeld, in Germany.

REFERENCES

[1] R. Agarwala, V. Bafna, M. Farach, B. Narayanan, M. Paterson, and M. Thorup, On the approximability of numerical taxonomy: fitting distances by tree metrics, Proceedings of the 7th Annual ACM-SIAM Symposium on Discrete Algorithms, 1996, pp. 365–372.

[2] D.J. Aldous, "Probability distributions on cladograms," Discrete random structures, IMA Vol. in Mathematics and its Applications, Vol. 76, D.J. Aldous and R. Permantle (Editors), Springer-Verlag, Berlin/New York, 1995, pp. 1–18.

[3] N. Alon and J.H. Spencer, The probabilistic method, Wiley, New York, 1992.

[4] A. Ambainis, R. Desper, M. Farach, and S. Kannan, Nearly tight bounds on the learnability of evolution, Proc of the 1998 Foundations of Comp Sci, to appear.

[5] K. Atteson, The performance of neighbor-joining algorithms of phylogeny reconstruction, Proc COCOON 1997, Computing and Combinatorics, Third Annual International Conference, Shanghai, China, Aug. 1997, Lecture Notes in Computer Science, Vol. 1276, Springer-Verlag, Berlin/New York, pp. 101–110.

[6] H.-J. Bandelt and A. Dress, Reconstructing the shape of a tree from observed dissimilarity data, Adv Appl Math 7 (1986), 309–343.

[7] V. Berry and O. Gascuel, Inferring evolutionary trees with strong combinatorial evidence, Proc COCOON 1997, Computing and Combinatorics, Third Annual International Conference, Shanghai, China, Aug. 1997, Lecture Notes in Computer Science, Vol. 1276, Springer-Verlag, Berlin/New York, pp. 111–123.

[8] J.K.M. Brown, Probabilities of evolutionary trees, Syst Biol 43 (1994), 78–91.

[9] D.J. Bryant and M.A. Steel, Extension operations on sets of leaf-labelled trees, Adv Appl Math 16 (1995), 425–453.

[10] P. Buneman, "The recovery of trees from measures of dissimilarity," Mathematics in the archaeological and historical sciences, F.R. Hodson, D.G. Kendall, P. Tatu (Editors), Edinburgh Univ. Press, Edinburgh, 1971, pp. 387–395.

[11] M. Carter, M. Hendy, D. Penny, L.A. Székely, and N.C. Wormald, On the distribution of lengths of evolutionary trees, SIAM J Disc Math 3 (1990), 38–47.

[12] J.A. Cavender, Taxonomy with confidence, Math Biosci 40 (1978), 271–280.

[13] J.T. Chang and J.A. Hartigan, Reconstruction of evolutionary trees from pairwise distributions on current species, Computing Science and Statistics: Proc 23rd Symp on the Interface, 1991, pp. 254–257.

[14] H. Colonius and H.H. Schultze, Tree structure for proximity data, British J Math Stat Psychol 34 (1981), 167–180.

[15] W.H.E. Day, Computational complexity of inferring phylogenies from dissimilarities matrices, Inform Process Lett 30 (1989), 215–220.

[16] W.H.E. Day and D. Sankoff, Computational complexity of inferring phylogenies by compatibility, Syst Zoology 35 (1986), 224–229.

[17] M.C.H. Dekker, Reconstruction methods for derivation trees, Master's Thesis, Vrije Universiteit, Amsterdam, 1986.

[18] P. Erdős and A. Rényi, On a classical problem in probability theory, Magy Tud Akad Mat Kutató Int Közl 6 (1961), 215–220.

[19] P.L. Erdős, M.A. Steel, L.A. Székely, and T. Warnow, Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule, Comput Artif Intell 16(2) (1997), 217–227.

[20] P.L. Erdős, M.A. Steel, L.A. Székely, and T. Warnow, "Inferring big trees from short quartets," ICALP'97, 24th International Colloquium on Automata, Languages, and Programming (Silver Jubilee of EATCS), Bologna, Italy, July 7–11, 1997, Lecture Notes in Computer Science, Vol. 1256, Springer-Verlag, Berlin/New York, 1997, 1–11.

[21] P.L. Erdős, M.A. Steel, L.A. Székely, and T. Warnow, A few logs suffice to build (almost) all trees-II, Theoret Comput Sci special issue on selected papers from ICALP 1997, to appear.

[22] P.L. Erdős, K. Rice, M. Steel, L. Szekely, and T. Warnow, The short quartet method, Mathematical Modeling and Scientific Computing, to appear.

[23] M. Farach and S. Kannan, Efficient algorithms for inverting evolution, Proc ACM Symp on the Foundations of Computer Science, 1996, pp. 230–236.

[24] M. Farach, S. Kannan, and T. Warnow, A robust model for inferring optimal evolutionary trees, Algorithmica 13 (1995), 155–179.

[25] J.S. Farris, A probability model for inferring evolutionary trees, Syst Zoology 22 (1973), 250–256.

[26] J. Felsenstein, Cases in which parsimony or compatibility methods will be positively misleading, Syst Zoology 27 (1978), 401–410.

[27] E.F. Harding, The probabilities of rooted tree shapes generated by random bifurcation, Adv Appl Probab 3 (1971), 44–77.

[28] M.D. Hendy, The relationship between simple evolutionary tree models and observable sequence data, Syst Zoology 38(4) (1989), 310–321.

[29] D. Hillis, Approaches for assessing phylogenetic accuracy, Syst Biol 44 (1995), 3–16.

[30] D. Hillis, Inferring complex phylogenies, Nature 383(12) (Sept. 1996), 130–131.

[31] D. Hillis, J. Huelsenbeck, and D. Swofford, Hobgoblin of phylogenetics? Nature 369 (1994), 363–364.

[32] M. Hendy, C. Little, and D. Penny, Comparing trees with pendant vertices labelled, SIAM J Appl Math 44 (1984), 1054–1065.

[33] J. Huelsenbeck, Performance of phylogenetic methods in simulation, Syst Biol 44 (1995), 17–48.

[34] J.P. Huelsenbeck and D. Hillis, Success of phylogenetic methods in the four-taxon case, Syst Biol 42 (1993), 247–264.

[35] S. Kannan, personal communication.

[36] M. Kimura, Estimation of evolutionary distances between homologous nucleotide sequences, Proc Nat Acad Sci USA 78 (1981), 454–458.

[37] J. Neyman, "Molecular studies of evolution: a source of novel statistical problems," Statistical decision theory and related topics, S.S. Gupta and J. Yackel (Editors), Academic Press, New York, 1971, pp. 1–27.

[38] H. Philippe and E. Douzery, The pitfalls of molecular phylogeny based on four species, as illustrated by the cetacea/artiodactyla relationships, J Mammal Evol 2 (1994), 133–152.

[39] K. Rice and T. Warnow, "Parsimony is hard to beat!," Proc COCOON 1997, Computing and combinatorics, Third Annual International Conference, Shanghai, China, Aug. 1997, Lecture Notes in Computer Science, Vol. 1276, Springer-Verlag, Berlin/New York, pp. 124–133.

[40] N. Saitou and M. Nei, The neighbor-joining method: A new method for reconstructing phylogenetic trees, Mol Biol Evol 4 (1987), 406–425.

[41] N. Saitou and T. Imanishi, Relative efficiencies of the Fitch–Mzargoliash, maximum parsimony, maximum likelihood, minimum evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree, Mol Biol Evol 6 (1989), 514–525.

[42] Y.S. Smolensky, A method for linear recording of graphs, USSR Comput Math Phys 2 (1969), 396–397.

[43] M.A. Steel, The complexity of reconstructing trees from qualitative characters and subtrees, J Classification 9 (1992), 91–116.

[44] M.A. Steel, Recovering a tree from the leaf colourations it generates under a Markov model, Appl Math Lett 7 (1994), 19–24.

[45] M.A. Steel, L.A. Székely, and P.L. Erdős, The number of nucleotide sites needed to accurately reconstruct large evolutionary trees, DIMACS Technical Report No. 96-19.

[46] M.A. Steel, L.A. Székely, and M.D. Hendy, Reconstructing trees when sequence sites evolve at variable rates, J Comput Biol 1 (1994), 153–163.

[47] K. Strimmer and A. von Haeseler, Quartet puzzling: a quartet maximum likelihood method for reconstructing tree topologies, Mol Biol Evol 13 (1996), 964–969.

[48] K. Strimmer, N. Goldman, and A. von Haeseler, Bayesian probabilities and quartet puzzling, Mol Biol Evol 14 (1997), 210–211.

[49] D.L. Swofford, G.J. Olsen, P.J. Waddell, and D.M. Hillis, "Phylogenetic inference," Molecular systematics, D.M. Hillis, C. Moritz, and B.K. Mable (Editors), Chap. 11, 2nd ed., Sinauer Associates, Inc., Sunderland, 1996, pp. 407–514.

[50] N. Takezaki and M. Nei, Inconsistency of the maximum parsimony method when the rate of nucleotide substitution is constant, J Mol Evol 39 (1994), 210–218.

[51] T. Warnow, Combinatorial algorithms for constructing phylogenetic trees, Ph.D. thesis, University of California-Berkeley, 1991.

[52] P. Winkler, personal communication.

[53] K.A. Zaretsky, Reconstruction of a tree from the distances between its pendant vertices, Uspekhi Math Nauk (Russian Math Surveys), 20 (1965), 90–92 (in Russian).

[54] A. Zharkikh and W.H. Li, Inconsistency of the maximum-parsimony method: The case of five taxa with a molecular clock, Syst Biol 42 (1993), 113–125.

[55] S.J. Wilson, Measuring inconsistency in phylogenetic trees, J Theoret Biol 190 (1998), 15–36.

ELSEVIER

Theoretical Computer Science 221 (1999) 77–118

Theoretical Computer Science

www.elsevier.com/locate/tcs

A few logs suffice to build (almost) all trees: Part II

Péter L. Erdős[a,*], Michael A. Steel[b], László A. Székely[c], Tandy J. Warnow[d]

[a] *Mathematical Institute of the Hungarian Academy of Sciences, P.O.Box 127,1364 Budapest, Hungary*
[b] *Biomathematics Research Centre, University of Canterbury, Christchurch, New Zealand*
[c] *Department of Mathematics, University of South Carolina, Columbia, SC, USA*
[d] *Department of Computer and Information Science University of Pennsylvania, Philadelphia, PA, USA*

Abstract

Inferring evolutionary trees is an interesting and important problem in biology, but one that is computationally difficult as most associated optimization problems are NP-hard. Although many methods are provably statistically consistent (i.e. the probability of recovering the correct tree converges to 1 as the sequence length increases), the actual rate of convergence for different methods has not been well understood. In a recent paper we introduced a new method for reconstructing evolutionary trees called the dyadic closure method (DCM), and we showed that DCM has a very fast convergence rate. DCM runs in $O(n^5 \log n)$ time, where n is the number of sequences, and so, although polynomial, the computational requirements are potentially too large to be of use in practice. In this paper we present another tree reconstruction method, the witness–antiwitness method (WAM). WAM is faster than DCM, especially on random trees, and converges to the true tree topology at the same rate as DCM. We also compare WAM to other methods used to reconstruct trees, including Neighbor Joining (possibly the most popular method among molecular biologists), and new methods introduced in the computer science literature.

Keywords: Phylogeny; Evolutionary tree reconstruction; Distance-based methods; Quartet methods; Short quartet methods; Dyadic closure method; Witness–antiwitness method

1. Introduction

Rooted leaf-labelled trees are a convenient way to represent historical relationships between extant objects, particularly in evolutionary biology (where such trees are called

* Corresponding author.
E-mail addresses: elp@math-inst.hu (P.L. Erdős), m.steel@math.canterbury.ac.nz. (M.A. Steel), laszlo@math.sc.edu. (L.A. Székely), tandy@central.cis.upenn.edu. (T.J. Warnow)

0304-3975/99/$ - see front matter
PII: S0304-3975(99)00028-6

"phylogenies"). Molecular techniques have recently provided large amounts of sequence (DNA, RNA, or amino-acid) data that are being used to reconstruct such trees. Statistically based methods construct trees from sequence data, by exploiting the variation in the sequences due to random mutations that have occurred. A typical assumption made by these tree construction methods is that the evolutionary process operates through "point mutations", where the positions, or "sites", within the sequences mutate down the tree. Thus, by modelling how the different sites evolve down the tree, the entire mutational process on the sequences can be described. A further assumption that is typically made is that the evolutionary processes governing each site are identical, and independent (i.i.d.). For such models of evolution, some tree construction methods are guaranteed to recover the underlying *unrooted* tree from adequately long sequences generated by the tree, with arbitrarily high probability.

There are two basic types of tree reconstruction methods: *sequence-based methods* and *distance-based methods*. Distance-based methods for tree reconstruction have two steps. In the first step, the input sequences are represented by an $n \times n$ matrix d of pairwise dissimilarities (these may or may not observe the triangle inequality, and hence may not be truly "distances"). In the second step, the method M computes an additive matrix $M(d)$ (that is, an $n \times n$ distance matrix which exactly fits an edge-weighted tree) from the pairwise dissimilarity matrix, d. Distance methods are typically polynomial time. Sequence-based methods, on the other hand, do not represent the relationship between the sequences as a distance matrix; instead, these methods typically attempt to solve NP-hard optimization problems based upon the original sequence data, and are computationally intensive. See [26] for further information on phylogenetic methods in general.

A tree reconstruction method, whether sequence-based or distance-based, is considered to be accurate with respect to the topology prediction if the tree associated (uniquely) with the computed additive matrix has the same unrooted topology as the tree used to generate the observed sequences. A method is said to be *statistically consistent* for a model tree T if the probability of recovering the topology of T from sequences generated randomly on T converges to 1 as the sequence length increases to infinity. It has long been understood that most distance-based methods are statistically consistent methods for inferring trees under models of evolution in which the sites evolve i.i.d., but that some sequence-based methods (notably, the optimization problem *maximum parsimony* [25]) are not statistically consistent on all trees under these models. For this reason, some biologists prefer to use distance-based methods. However, not much is known, even experimentally, about the sequence length a given distance-based method needs for exact topological accuracy with high probability. How long the sequences have to be to guarantee high probability of recovering the tree depends on the reconstruction method, the details of the model, and the number n of species. Determining bounds on that length and its growth with n has become more pressing since biologists have begun to reconstruct trees on increasingly larger numbers of species (often up to several hundred) from such sequences.

In a previous paper [20], we addressed this question for trees under the Neyman 2-state model of site evolution, and obtained the following results:

1. We established a lower bound of $\log n$ on the sequence length that every method, randomized or deterministic, requires in order to reconstruct any given n-leaf tree in any 2-state model of sequence evolution,
2. We showed that the maximum compatibility method of phylogenetic tree construction requires sequences of length *at least* $n \log n$ to obtain the tree with high probability, and
3. We presented a new polynomial time method (the *dyadic closure method* (DCM)) for reconstructing trees in the Neyman 2-state model, and showed that polylogarithmic length sequences suffice for accurate tree reconstruction with probability near one on almost all trees, and polynomial length sequence length always suffices for any tree under reasonable assumptions on mutation probabilities.

Thus, the DCM [20] has a very fast convergence rate, which on almost all trees is within a polynomial of our established lower bound of $\log n$ for any method. However, although DCM uses only polynomial time, it has large computational requirements (it has $\Omega(n^2k + n^5 \log n)$ running time, and uses $O(n^4)$ space), where k is the sequence length. This may make it infeasible for reconstructing large trees.

In this paper, we present the *witness–antiwitness method* (WAM), a new and faster quartet-based method for tree reconstruction, which has the same asymptotic convergence rate as the DCM. The running time of WAM has a worst-case bound $O(n^2k + n^4 \log n \log k)$ where k is the sequence length, and is even faster under some reasonable restrictions on the model (see Theorem 12 for details). Thus, WAM is a faster algorithm than DCM, and has essentially the same convergence rate to the true tree topology as DCM. The *provable* bounds on the running time of WAM depend heavily on the depth of the model tree. We introduced the "depth" in [20] and showed that $depth(T)$ is bounded from above by $\log n$ for all binary trees T, and that random trees have depths bounded by $O(\log \log n)$.

In addition to presenting the new method, we present a framework for a comparative analysis of the convergence rates of different distance based methods. We apply this technique to several different methods, *neighbor joining* [43], the Agarwala et al. [1] "single-pivot" algorithm and its variant [21], the "double-pivot" algorithm, and the naive quartet method (a method we describe in this paper). We obtain *upper bounds* on the sequence lengths that suffice for accuracy for these distance-based methods, and show that these upper bounds grow exponentially in the *weighted diameter* of the tree, which is the maximum number of expected mutations for a random site on any leaf-to-leaf path in the tree. We analyze the weighted diameter of random trees under two distributions. We show that the diameter of random trees is $\Omega(\sqrt{n})$ under the uniform distribution, and $\Omega(\log n)$ under the Yule–Harding distribution. Consequently, these upper bounds on the sequence lengths that suffice for accuracy for these other distance-based methods are significantly larger than the upper bounds obtained for DCM and WAM. We note that our upper bounds for the algorithms in [1, 21] match those given by Sampath Kannan (personal communication). Finally, we generalize our methods and

results to more general Markov models, and find the same relative performance (these results should be compared to those of Ambainis et al. in [4]). (While this framework provides a comparison between the convergence rates of these methods, it is limited by the fact that these are *upper bounds* on the sequence lengths that suffice for accuracy for these distance methods. These upper bounds may be loose, but no better upper bounds on these methods are yet known, to our knowledge. Obtaining better bounds on the convergence rates of these and other methods is an important open question.)

The structure of the paper is as follows. In Section 2 we provide definitions and discuss tree reconstruction methods in general. In Section 3, we describe the analytical framework for deriving upper bounds on the sequence lengths needed by different methods for exact accuracy in tree reconstruction, and we use this framework to provide an initial comparison between various distance-based methods. In Section 4, we describe the witness–antiwitness tree construction algorithm (WATC), and in Section 5, we describe the witness–antiwitness method (WAM) in full. In Section 6, we analyze the performance of WAM for reconstructing trees under the Neyman model of site evolution, and compare its performance to other promising distance-based methods. We extend the analysis of WAM to reconstructing trees under the general r-state Markov model in Section 7. Finally, in Section 8, we disucss the applicability of our results to biological data.

2. Definitions

Notation. $\mathbb{P}[A]$ denotes the probability of event A; $\mathbb{E}[X]$ denotes the expectation of random variable X. We denote the natural logarithm by log. The set $[n]$ denotes $\{1,2,\ldots,n\}$ and for any set S, $\binom{S}{k}$ denotes the collection of subsets of S of size k. $\mathbb{R}$ denotes the real numbers.

Definition. (I) *Trees.* We will represent a phylogenetic tree T by a *semi-labelled* tree whose *leaves* (vertices of degree one) are labelled by extant species, numbered by $1,2,\ldots,n$, and whose remaining internal vertices (representing ancestral species) are unlabelled. We will adopt the biological convention that phylogenetic trees are *binary*, meaning that all internal nodes have degree three, and we will also assume that T is *unrooted* (this is due to scientific and technical reasons which indicate that the location of the root can be either difficult or impossible to determine from data). We let $B(n)$ denote the set of all $(2n-5)!! = (2n-5)(2n-7)\cdots 3\cdot 1$ semi-labelled binary trees on the leaf set $[n]$.

The path between vertices u and v in the tree is called the *uv path*, and is denoted $P(u,v)$. The *topological distance* $L(u,v)$ between vertices u and v in a tree T is the number of edges in $P(u,v)$. The edge set of the tree is denoted by $E(T)$. Any edge adjacent to a leaf is called a *leaf edge*, any other edge is called an *internal edge*. For a phylogenetic tree T and $S \subseteq [n]$, there is a unique minimal subtree of T, containing all elements of S. We call this tree the *subtree* of T induced by S, and denote it by

$T_{|S}$. We obtain the *contracted subtree induced by* S, denoted by $T^*_{|S}$, if we substitute edges for all maximal paths of $T_{|S}$ in which every internal vertex has degree two. We denote by $ij|kl$ the tree on four leaves i,j,k,l in which the pair i,j is separated from the pair k,l by an internal edge. When the contracted subtree of T induced by leaves i,j,k,l is the tree $ij|kl$, we call $ij|kl$ a *valid quartet split* of T on the quartet of leaves $\{i,j,k,l\}$. Since all trees are assumed to be binary, all contracted subtrees (including, in particular, the quartet subtrees) are also binary. Consequently, the set $Q(T)$ of valid quartet splits for a binary tree T has cardinality $\binom{n}{4}$.

(II) *Sites.* Consider a set C of character states (such as $C=\{A,C,G,T\}$ for DNA sequences; $C=\{$the 20 amino acids$\}$ for protein sequences; $C=\{R,Y\}$ or $\{0,1\}$ for purine–pyrimidine sequences). A *sequence of length* k is an ordered k-tuple from C – that is, an element of C^k. A collection of n such sequences – one for each species labelled from $[n]$ – is called a *collection of aligned sequences.*

Aligned sequences have a convenient alternative description as follows. Place the aligned sequences as rows of an $n\times k$ matrix, and call *site* i the ith column of this matrix. A *pattern* is one of the $|C|^n$ possible columns.

(III) *Site substitution models.* Many models have been proposed to describe the evolution of sites as a stochastic process. Such models depend on the underlying phylogenetic tree T and some randomness. Most models assume that the sites are independently and identically distributed (i.i.d.).

The models on which we test our algorithm also assume the Markov property that the random assignment of a character state to a vertex v is determined by the character state of its immediate ancestor, and a random substitution on the connecting edge. In the most general stochastic model that we study, the sequence sites evolve i.i.d. according to the general Markov model from the root [47]. We now briefly discuss this general Markov model. Since the i.i.d. condition is assumed, it is enough to consider the evolution of a single site in the sequences. Substitutions (point mutations) at a site are generally modelled by a probability distribution π on a set of $r>1$ character states at the root ρ of the tree (an arbitrary vertex or a subdividing point on an edge), and each edge e oriented out from the root has an associated $r\times r$ stochastic transition matrix $M(e)$. The random character state at the root "evolves" down the tree – thereby assigning characters randomly to the vertices, from the root down to the leaves. For each edge $e=(u,v)$, with u between v and the root, $(M(e))_{\alpha\beta}$ is the probability that v has character state β given that u has character state α.

(IV) *The Neyman model.* The simplest stochastic model is a symmetric model for binary characters due to Neyman [40], and was also developed independently by Cavender [12] and Farris [24]. Let $\{0,1\}$ denote the two states. The root is a fixed leaf, the distribution π at the root is uniform. For each edge e of T we have an associated *mutation probability*, which lies strictly between 0 and 0.5. Let $p:E(T)\to(0,0.5)$ denote the associated map. We have an instance of the general Markov model with $M(e)_{01}=M(e)_{10}=p(e)$. We will call this the *Neyman 2-state model*, but note that it has also been called the Cavender–Farris model, and is equivalent to the Jukes–Cantor model when restricted to two states.

The Neyman 2-state model is hereditary on subsets of the leaves – that is, if we select a subset S of $[n]$, and form the subtree $T_{|S}$, then eliminate vertices of degree two, we can define mutation probabilities on the edges of $T^*_{|S}$ so that the probability distribution on the patterns on S is the same as the marginal of the distribution on patterns provided by the original tree T. Furthermore, the mutation probabilities that we assign to an edge of $T^*_{|S}$ is just the probability p that the endpoints of the associated path in the original tree T are in different states.

Lemma 1. *The probability p that the endpoints of a path P of topological length k are in different states is related to the mutation probabilities $p_1, p_2, \ldots, p_k$ of edges of P as follows*:

$$p = \frac{1}{2}\left(1 - \prod_{i=1}^{k}(1-2p_i)\right).$$

Lemma 1 is folklore and is easy to prove by induction.

(V) *Distances.* Any symmetric matrix, which is zero-diagonal and positive off-diagonal, will be called a *distance matrix*. (These "distances", however, may not satisfy the triangle inequality, because the distance corrections used in phylogenetics, and described below, do not always satisfy the triangle inequality. Since it is nevertheless the practice in systematics to refer to these quantities as "distances", we will do so here as well.) An $n \times n$ distance matrix D_{ij} is called *additive*, if there exists an n-leaf tree (not necessarily binary) with positive edge lengths on the internal edges and non-negative edge lengths on the leaf edges, so that D_{ij} equals the sum of edge lengths in the tree along the $P(i,j)$ path connecting leaves i and j. In [10], Buneman showed that the following four-point condition characterizes additive matrices (see also [45, 64]):

Theorem 1 (Four-point condition). *A matrix D is additive if and only if for all i,j,k,l (not necessarily distinct), the maximum of $D_{ij}+D_{kl}$, $D_{ik}+D_{jl}$, $D_{il}+D_{jk}$ is not unique. The tree with positive lengths on internal edges and non-negative lengths on leaf edges representing the additive distance matrix is unique among the trees without vertices of degree two.*

Given a pair of parameters (T, p) for the Neyman 2-state model, and sequences of length k generated by the model, let $H(i,j)$ denote the *Hamming distance* of sequences i and j and $h^{ij} = H(i,j)/k$ denote the *dissimilarity score* of sequences i and j. The *empirical corrected distance* between i and j is denoted by

$$d_{ij} = -\tfrac{1}{2}\log(1-2h^{ij}). \tag{1}$$

The probability of a change in the state of any fixed character between the sequences i and j is denoted by $E^{ij} = \mathbb{E}(h^{ij})$, and we let

$$D_{ij} = -\tfrac{1}{2}\log(1-2E^{ij}) \tag{2}$$

denote the *corrected model distance* between i and j. We assign to any edge e a positive length

$$l(e) = -\tfrac{1}{2}\log(1-2p(e)). \tag{3}$$

By Lemma 1, D_{ij} is the sum of the lengths (see previous equation) along the path $P(i,j)$ between i and j, and hence D_{ij} is an additive distance matrix. Furthermore, d_{ij} converges in probability to D_{ij} as the sequence length tends to infinity. These mathematical facts also have significance in biology, since under certain continuous time Markov models [48], which may be used to justify our models, $l(e)$ and D_{ij} are the expected number of back-and-forth state changes along edges and paths, respectively. A similar phenomenon and hence a similar distance correction exists for the general stochastic model [47], and is discussed in detail in Section 7.

(VI) *Tree reconstruction.* A *phylogenetic tree reconstruction method* is a function Φ that associates either a tree or the statement *Fail* to every collection of aligned sequences, the latter indicating that the method is unable to make such a selection for the data given.

According to the practice in systematic biology (see, for example, [31, 32, 52]), a method is considered to be *accurate* if it recovers the unrooted binary tree T, even if it does not provide any estimate of the mutation probabilities. A necessary condition for accuracy, under the models discussed above, is that two distinct trees, T, T', do not produce the same distribution of patterns no matter how the trees are rooted, and no matter what their underlying Markov parameters are. This "identifiability" condition is violated under an extension of the i.i.d. Markov model when there is an unknown distribution of rates across sites as described by Steel et al. [49]. However, it is shown in [47] (see also [13]) that the identifiability condition holds for the i.i.d model under the weak conditions that the components of π are not zero and, for each edge e, the determinant $\det(M(e)) \neq 0, 1, -1$, and in fact we can recover the underlying tree from the expected frequencies of patterns on just *pairs* of species.

Theorem 1 and the discussion that follows it suggest that appropriate methods applied to corrected distances will recover the correct tree topology from sufficiently long sequences. Consequently, one approach (which is guaranteed to yield a *statistically consistent* estimate) to reconstructing trees from distances is to seek an additive distance matrix of minimum distance (with respect to some metric on distance matrices) from the input distance matrix. Many metrics have been considered, but all resultant optimization problems have been shown or are assumed to be NP-hard (see [1, 17, 23] for results on such problems).

(VII) *Specific tree construction algorithms.* In this paper, we will be particularly interested in certain distance methods, the *four-point method* (FPM), the *naive method*, *neighbor joining*, and the *Agarwala* et al. algorithm. We now describe these methods.

Four-Point Method (FPM). Given a 4×4 distance matrix d, return the split $ij|kl$ which satisfies $d_{ij} + d_{kl} < \min\{d_{ik} + d_{jl}, d_{il} + d_{jk}\}$. If there is no such split, return *Fail*.

FPM is a not truly a tree reconstruction method, because it can only be applied to datsets of size four. We include it here, because it is a subroutine in the Naive Method, which we now describe.

The *Naive Method* uses the four-point method to infer a split for every quartet i,j,k,l. Thus, if the matrix is additive, the four-point method can be used to detect the valid quartet split on every quartet of vertices, and then standard algorithms [6, 14] can be used to reconstruct the tree from the set of splits. Note that the naive method is guaranteed to be accurate when the input distance matrix is *additive*, but it will also be accurate even for non-additive distance matrices under conditions which we will describe later (see Section 3). Most quartet-based methods (see, for example, [7, 50, 51]) begin in the same way, constructing a split for every quartet, and then accommodate possible inconsistencies using some technique specific to the method; the naive method, by contrast, only returns a tree if all inferred splits are consistent with that tree. The obvious optimization problem (find a maximum number of quartets which are simultaneously realizable) is of unknown computational complexity.

The *Agarwala et al.* algorithm [1] is a 3-approximation algorithm for the nearest tree with respect to the L_∞-metric, where $L_\infty(A,B) = \max_{ij} |A_{ij} - B_{ij}|$. Given input d, the result of applying the Agarwala et al. algorithm to d is an additive distance matrix D such that $L_\infty(d,D) \leqslant 3L_\infty(d,D^{\text{opt}})$, where D^{opt} is an optimal solution.

The use of the Agarwala et al. algorithm for inferring trees has been studied in two papers (see [22] for a study of its use for inferring trees under the Neyman model, and [4] for a study of its use for inferring trees under the general Markov model). However, both [22, 4] consider the performance of the Agarwala et al. algorithm with respect to the *variational distance* metric. Optimizing with respect to this metric is related to – but distinct from – estimating the tree T, since it is concerned as well with the mutational parameters p.

The *neighbor joining* method [43] is a method for reconstructing trees from distance matrices, which is based upon agglomerative clustering. It is possibly the most popular method among molecular biologists for reconstructing trees, and does surprisingly well in some experimental studies; see, for example, [34, 35].

All these methods are known to be statistically consistent for inferring trees both under the Neyman 2-state model and under the general r-state Markov model of site evolution.

3. A framework for the comparison of distance-based methods

Although it is understood that all reasonable distance-based methods will converge on the true tree given sequences of adequate length, understanding the rate of convergence (as a function of sequence length) to the true topology is more complicated. However, it is possible sometimes to compare different distance-based methods, without reference to the underlying model. The purpose of this section is to provide a framework for an explicit comparison among different distance-based methods. We will use

this technique to compare the 3-approximation algorithm of Agarwala et al. to the *Naive method.* Our analysis of these two algorithms shows that on any distance matrix for which the first algorithm is guaranteed to reconstruct the true tree, so is the naive method. Since our new method, WAM, is guaranteed to reconstruct the true tree on any dataset for which the naive method is also guaranteed to reconstruct the true tree, this analysis also establishes a comparison between the Agarwala et al. algorithm and WAM.

By the four-point condition (Theorem 1) every additive distance matrix corresponds to a unique tree without vertices of degree 2, and with positive internal edge lengths, and non-negative lengths on edges incident with leaves.

Suppose we have a binary model tree T with positively weighted internal edges. Let x be the minimum edge-weight among internal edges, and let D be the associated additive distance matrix. Let d be an observed distance matrix, and let $\Delta = L_\infty(d,D)$.

For every distance-based reconstruction method Φ, we seek a constant $c(\Phi)$ such that

$$c(\Phi) = \sup\{c\colon\ \Delta < cx \Rightarrow \Phi(d) \text{ yields } T\}.$$

Lemma 2. (i) *Two additive distance matrices D and D' define the same topology if and only if for all quartets the relative orders of the pairwise sums of distances for that quartet are identical in the two matrices.*

(ii) *For every edge-weighted binary tree T with minimum internal edge weight x, and any $\vartheta > 0$, there is a different binary tree T' such that $L_\infty(D,D') = x/2 + \vartheta$, where D' is the additive distance matrix for T'.*

(iii) *Given any $n \times n$ distance matrix d, four indices i,j,k,l in $[n]$, let p_{ijkl} denote the difference between the maximum and the median of the three pairwise sums, $d_{ij} + d_{kl}$, $d_{ik} + d_{jl}$, $d_{il} + d_{jk}$. Let P be the maximum of the p_{ijkl} over all quartets i,j,k,l. Then there is no additive distance matrix D such that $L_\infty(d,D) < P/4$.*

Proof. Claim (i) is a direct consequence of the four-point condition (Theorem 1).

To prove (ii), for a given T, contract an internal edge e having minimum edge weight x, obtaining a non-binary tree T'. T' has exactly one vertex adjacent to four edges. Add $x/4$ to the weight of each of the four edges. Insert a new edge of weight ϑ to resolve the vertex of degree four, so that we obtain a binary tree T'', different from T. Let D be the additive distance matrix for T and let D'' be the additive distance matrix for T''. It is easy to see that then $L_\infty(D,D') = x/2 + \vartheta$.

For the proof of (iii), let D be an additive distance matrix with $L_\infty(d,D) = \varepsilon < t/4$. For all quartets i,j,k,l, the median and the maximum of the three pairwise sums induced by i,j,k,l are identical in D. Now consider the quartet i,j,k,l for which $p_{ijkl} = t$. The maximum and the median of the three pairwise sums in d differ by p_{ijkl}. In order for the maximum and median of the three pairwise sums to be equal in D, at least one pairwise distance must change by at least $p_{ijkl}/4$. However $\varepsilon < p_{ijkl}/4$, contradicting the assumption. □

Theorem 2. *Let D be an additive $n \times n$ distance matrix defining a binary tree T, d be a fixed distance matrix, and let $\delta = L_\infty(d, D)$. Assume that x is the minimum weight of internal edges of T in the edge weighting corresponding to D.*

(i) *A hypothetical exact algorithm for the L_∞-nearest tree is guaranteed to return the topology of T from d if $\delta < x/4$.*

(ii) (a) *The 3-approximation algorithm for the L_∞-nearest tree is guaranteed to return the topology of T from d if $\delta < x/8$.* (b) *For all n there exists at least one d with $\delta = x/6$ for which the method can err.* (c) *If $\delta \geqslant x/4$, the algorithm can err for every such d.*

(iii) *The naive method is guaranteed to return the topology of T from d if $\delta < x/2$, and there exists a d for any $\delta > x/2$ for which the method can err.*

Proof. To prove (i), assume that D^* is an additive distance matrix with $L_\infty(d, D^*) \leqslant \delta$, and let T^* denote the tree topology corresponding to D^*. According to Lemma 2, Part (i), D^* and D define the same tree iff the relative order of pairwise sums of distances agree for all quartets in the two matrices. We will prove that D^* and D define the same tree topology by contradiction.

So suppose D^* and D do not define the same tree topology. Then there is a quartet, i, j, k, l, of leaves, where (without loss of generality) the topology induced by matrix D is $ij|kl$ and the topology induced by matrix D^* is $ik|jl$. Thus, there exist positive constants P and ε so that $2P + D_{ij} + D_{kl} = D_{ik} + D_{jl}$ and $D^*_{ij} + D^*_{kl} = D^*_{ik} + D^*_{jl} + 2\varepsilon$. Now $P \geqslant x$, since P is an internal path length in T. By the triangle inequality we have

$$L_\infty(D, D^*) \leqslant 2\delta. \tag{4}$$

We have

$$2P + 2\varepsilon = D_{ik} + D_{jl} - D_{ij} - D_{kl} + D^*_{ij} + D^*_{kl} - D^*_{ik} - D^*_{jl} \tag{5}$$

and hence by the triangle inequality

$$2x < 2P + 2\varepsilon \leqslant 8\delta. \tag{6}$$

Since $\delta < x/4$, this implies that such a quartet i, j, k, l does not exist, and so D and D^* define the same tree topology.

To prove (ii)(a), let D^* denote the output of the 3-approximation algorithm and T^* denote the corresponding tree. Following similar arguments, $L_\infty(d, D^*) \leqslant 3\delta$, so that corresponding to formula (4) we have $L_\infty(D, D^*) \leqslant 4\delta$, and corresponding to formula (6) we have $2x < 16\delta$. To prove (ii)(b), we now give an example where the 3-approximation algorithm can fail in which $L_\infty(D, d) = x/6$. Let d be distance matrix defined by $d_{uv} = d_{wx} = 7/3$, $d_{uw} = d_{vx} = 3$ and $d_{ux} = d_{vw} = 10/3$. By item (iii) of Lemma 2, it follows that there is no additive distance matrix D with $L_\infty(d, D) < 1/6$. Now let D be the additive distance matrix induced by the binary tree T on leaves u, v, w, x with topology $uv|wx$ and with edge length as follows: the central edge in T has weight 1 and all other edges have weight 13/12. Then, $L_\infty(D, d) = 1/6$ so that D

is a closest additive distance matrix to d. Furthermore, $L_\infty(d,D)=x/6$, since $x=1$ is the lowest edge weight in T. However there is another additive distance matrix induced by a different tree which lies within 3 times this minimal distance. Namely, let D'' be the additive distance matrix induced by the binary tree with topology $uw|vx$ with interior edge weighted 1/3 and other edges weighted 5/4. Then, $L_\infty(D'',d)=1/2=3L_\infty(D,d)=3\min_D\{L_\infty(D,d)\}$, as claimed. It is easy to see that this example can be embedded in any size distance matrix so that for all n such examples exist. For (ii)(c), suppose d is a distance matrix, D is its closest additive distance matrix, and x is the smallest weight of any edge in D. Then contract the edge e of weight x in T, the edge-weighted realization of D, and add $x/4$ to every edge originally incident to e. Let D' be the distance matrix of the new edge-weighted tree, T'. It follows that $L_\infty(D,D')=x/2$ and so that $L_\infty(d,D')\leqslant L_\infty(d,D)+L_\infty(D,D')$. If $L_\infty(d,D)=x/4$, then $L_\infty(d,D')\leqslant 3x/4$, by the triangle inequality. Hence the 3-approximation algorithm could return the topology of T or of T', and since they are different there is a possibility of making the wrong choice.

To prove (iii), arguments similar to the ones above obtain

$$2P+2\varepsilon=D_{ik}+D_{jl}-D_{ij}-D_{kl}+d_{ij}+d_{kl}-d_{ik}-d_{jl}$$

and $2x<2P+2\varepsilon\leqslant 4\delta$. The required example is in Lemma 2, Part (ii). □

In other words, *given any matrix d of corrected distances, if an exact algorithm for the L_∞-nearest tree can be guaranteed – by this analysis – to correctly reconstruct the topology of the model tree, then so can the Naive method.* This may suggest that there is an inherent limitation of the L_∞-nearest tree approach to reconstructing phylogenetic tree topologies. However, note that the analytical results are pessimistic; that is, they guarantee a high probability of an accurate performance once sequence lengths exceed some threshold, but do not guarantee a low probability of accurate performance for sequences below those lengths. Even so, these techniques are essentially the same ones that have been used in other studies to obtain analytical results regarding convergence to the true tree (see also [4, 22]).

4. The witness–antiwitness tree construction (WATC)

4.1. Introduction

In this section we describe the witness–antiwitness tree construction algorithm (WATC). This procedure, which is the heart of our witness–antiwitness method (WAM), solves certain restricted instances of the NP-complete quartet consistency problem [46], and solves them faster than the dyadic closure tree construction algorithm (DCTC) that we used as a procedure previously in our dyadic closure method (DCM) [20]. We therefore achieve an improvement with respect to computational requirements over DCM, and pay for it by requiring somewhat longer sequences.

Let e be an edge in T. Deleting e but not its endpoints creates two rooted subtrees, T_1 and T_2; these are called *edi-subtrees*, where "edi" stands for "*e*dge-*d*eletion-*i*nduced". Each edi-subtree having at least two leaves can be seen as being composed of two smaller edi-subtrees. The algorithm we will describe, the witness–antiwitness tree construction algorithm, or WATC, constructs the tree "from the outside in", by inferring larger and larger edi-subtrees, until the entire tree is defined. Thus, the algorithm has to decide at each iteration at least one pair of edi-subtrees to "join" into a new edi-subtree. In the tree, such pairs can be recognized by the constraints (a) that they are disjoint, and (b) that their roots are at distance two from each other. These pairs of edi-subtrees are then said to be "siblings". The algorithm determines whether a pair of edi-subtrees are siblings by using the quartet splits. We will show that if the set Q satisfies certain conditions then WATC is guaranteed to reconstruct the tree T from Q.

The conditions that Q must satisfy in order for WATC to be guaranteed to reconstruct the tree T are slightly more restrictive than those we required in the DCTC method, but do not require significantly longer sequences. Sets Q which satisfy these conditions are said to be *T-forcing*. The first stage of WATC assumes that Q is *T-forcing*, and on that basis attempts to reconstruct the tree T. If during the course of the algorithm it can be determined that Q is *not* T-forcing, then the algorithm returns *Fail*. Otherwise, a tree T' is constructed. At this point, the second stage of WATC begins, in which we determine whether T is the unique tree that is consistent with Q. If Q fails this test, then the algorithm returns *Fail*, and otherwise it returns T.

Just as in the dyadic closure method (DCM) we will need a search technique to find an appropriate set Q. Whereas binary search was a feasible technique for the DCM, it is no longer feasible in this case. Search techniques for an appropriate set Q are discussed in Section 5.

4.2. Definitions and preliminary material

Within each *edi*-subtree t, select that unique leaf which is the lowest valued leaf among those closest topologically to the root (recall that leaves are identified with positive integers). This is called the *representative* of t, and is denoted $rep(t)$. If the edi-subtree consists of a single leaf, then the representative leaf is identical with this single leaf, which also happens to be the root of the edi-subtree at the same time.

The *diameter* of the tree T, $diam(T)$, is the maximum topological distance in the tree between any pair of leaves. We define the depth of an edi-subtree t to be $L(root(t), rep(t))$, and denote this quantity by $depth(T)$. The *depth* of T is then $\max_t\{depth(t)\}$, as t ranges over all edi-subtrees yielded by internal edges of T. We say that a path P in the tree T is *short* if its topological length is at most $depth(T)+1$, and say that a quartet i,j,k,l is a *short quartet* if it induces a subtree which contains a single edge connected to four disjoint short paths. The set of all short quartets of the tree T is denoted by $Q_{\text{short}}(T)$. We will denote the set of valid quartet splits for the short quartets by $Q^*_{\text{short}}(T)$.

For each of the $n-3$ internal edges of the n-leaf binary tree T we assign a *representative quartet* $\{i,j,k,l\}$ as follows. The deletion of the internal edge *and* its endpoints defines four rooted subtrees. Pick the representative from each of these subtrees to obtain i,j,k,l; by definition, the quartet i,j,k,l is a *short quartet* in the tree. We call the split of this quartet a *representative quartet split* of T, and we denote the set of representative quartet splits of T by R_T. Note that by definition

$$R_T \subseteq Q^*_{\text{short}}(T) \subseteq Q(T). \tag{7}$$

We will say that a set Q of quartet splits is *consistent* with a tree T if $Q \subseteq Q(T)$. We will say that Q is *consistent* if there exists a tree T with which Q is consistent, and otherwise Q is said to be *inconsistent*. In [20], we proved:

Theorem 3. *Let T be a binary tree on $[n]$. If R_T is consistent with a binary tree T' on $[n]$, then $T = T'$. Therefore, if $R_T \subseteq Q$, then either Q is inconsistent, or Q is consistent with T. Furthermore, Q cannot be consistent with two distinct trees if $R_T \subseteq Q$.*

Let S be a set of n sequences generated under the Neyman model of evolution, and let d be the matrix of corrected empirical distances. Given any four sequences i,j,k,l from S, we define the *width* of the quartet on i,j,k,l to be $\max(d_{ij}, d_{ik}, d_{il}, d_{jk}, d_{jl}, d_{kl})$. For any $w \in \mathbb{R}^+$, let Q_w denote the set of quartet splits of width at most w, inferred using the four-point method.

4.3. The dyadic closure method

The dyadic closure method is based on the dyadic closure tree construction (DCTC) algorithm, which uses dyadic closure (see [20, 18]) to reconstruct a tree T consistent with an input set Q of quartet splits. Recall that $Q(T)$ denotes the set of all valid quartet splits in a tree T, and that given $Q(T)$, the tree T is uniquely defined. The *dyadic closure* of a set Q is denoted by $cl(Q)$, and consists of all splits that can be inferred by combining two splits at a time from Q, and from previously inferred quartet splits. In [20], we showed that the dyadic closure $cl(Q)$ could be computed in $O(n^5)$ time, and that if Q contained all the representative quartet splits of a tree, and contained only valid quartet splits, (i.e. if $R_T \subseteq Q \subseteq Q(T)$), then $cl(Q) = Q(T)$. Consequently, the DCTC algorithm reconstructs the tree T if $R_T \subseteq Q \subseteq Q(T)$. It is also easy to see that no set Q can simultaneously satisfy this condition for two distinct binary trees T, T', by Theorem 3, and furthermore, if Q satisfies this condition for T, it can be quickly verified that T is the unique solution to the reconstruction problem. Thus, when Q is such that for some binary tree T, $R_T \subseteq Q \subseteq Q(T)$, then the DCTC algorithm properly reconstructs T. The problem cases are when Q does not satisfy this condition for any T.

We handle the problem cases by specifying the output $DCTC(Q)$ to be as follows:

- *binary tree* T such that $cl(Q) = Q(T)$ (this type of output is guaranteed when $R_T \subseteq Q \subseteq Q(T)$),

- *inconsistent* when $cl(Q)$ contains two contradictory splits for the same quartet, or
- *insufficient* otherwise.

Note that this specification does not prohibit the algorithm from reconstructing a binary tree T, even if Q does not contain all of R_T. In such a case, the tree T will nevertheless satisfy $cl(Q)=Q(T)$; therefore, no other binary tree T' will satisfy $Q\subseteq Q(T'))$. Note that if $DCTC(Q)=Inconsistent$, then $Q\not\subseteq Q(T)$ for any binary tree T, so that if $Q\subseteq Q'$ then $DCTC(Q')=Inconsistent$ as well. On the other hand, if $DCTC(Q)=Insufficient$ and $Q'\subseteq Q$, then $DCTC(Q')=Insufficient$ also. Thus, if $DCTC(Q)$ is Inconsistent, then there is no tree T consistent with Q, but if $DCTC(Q)$ is Insufficient, then it is still possible that some tree exists consistent with Q, but the set Q is *insufficient* with respect to the requirements of the DCTC method.

Now consider what happens if we let Q be Q_w the set of quartet splits based upon quartets of width at most w. The output of the DCTC algorithm will indicate whether w is too big (i.e. when $DCTC(Q_w)=Inconsistent$), or too small (i.e. when $DCTC(Q_w)=Insufficient$). Consequently, DCTC can be used as part of a tree construction method, where splits of quartets (of some specified width w) are estimated using some specified method, and we search through the possible widths w using binary search.

In [20], we studied a specific variant of this approach, called the Dyadic Closure Method (DCM), in which quartet trees are estimated using the four-point method (see Definition VII in Section 2). We analyzed the sequence length that suffices for accurate tree construction by DCM and showed that it grows very slowly; for almost all trees under two distributions on binary trees the sequence length that suffices for tree reconstruction under DCM is only polylogarithmic in n, once $0<f\leqslant g<.5$ are fixed and $p(e)\in[f,g]$ is assumed. Thus, DCM has a very fast convergence rate. DCM uses $O(n^2k+n^5\log n)$ time and $O(n^4)$ space; therefore it is a statistically consistent polynomial time method for inferring trees under the Neyman model of evolution. For practical purposes, however, the computational requirements of the DCM method are excessive for inferring large trees, where n can be on the order of hundreds.

4.4. Witnesses, antiwitnesses, and T-forcing sets

Recall that the witness–antiwitness tree construction algorithm constructs T from the outside in, by determining in each iteration which pairs of edi-subtrees are siblings. This is accomplished by using the quartet splits to guide the inference of edi-subtrees. We now describe precisely how this is accomplished.

Definition 1. Recall that an *edi-subtree* is a subtree of T induced by the deletion of an edge in the tree. Two edi-subtrees are *siblings* if they are disjoint, the path between their roots contains exactly two edges, and there are at least two leaves not in either of these two edi-subtrees. (The last condition – that there are at least two leaves not in either of the two edi-subtrees – is nonstandard, but is assumed because it simplifies our discussion.) Let t_1 and t_2 be two vertex disjoint *edi*-subtrees. A *witness*

to the siblinghood of t_1 *and* t_2 is a quartet split $uv|wx$ such that $u \in t_1$, $v \in t_2$, and $\{w,x\} \cap (t_1 \cup t_2) = \emptyset$. We call such quartets *witnesses*. An *anti-witness to the siblinghood of* t_1 *and* t_2 is a quartet split $pq|rs$, such that $p \in t_1$, $r \in t_2$, and $\{q,s\} \cap (t_1 \cup t_2) = \emptyset$. We will call these *anti-witnesses*.

Definition 2. Let T be a binary tree and Q a set of quartet splits defined on the leaves of T.

- Q has the *witness property for* T: Whenever t_1 and t_2 are sibling edi-subtrees of T and $T - t_1 - t_2$ has at least two leaves, then there is a quartet split of Q which is a witness to the siblinghood of t_1 and t_2.
- Q has the *antiwitness property for* T: Whenever there is a witness in Q to the siblinghood of two edi-subtrees t_1 and t_2 which are not siblings in T, then there is a quartet split in Q which is an antiwitness to the siblinghood of t_1 and t_2.

Theorem 4. *If* $R_T \subseteq Q$, *then* Q *has the witness property for* T. *Furthermore, if* $R_T \subseteq Q \subseteq Q(T)$, *and* t_1 *and* t_2 *are sibling edi-subtrees, then* Q *contains at least one witness, but no antiwitness, to the siblinghood of* t_1 *and* t_2.

The proof is straightforward, and is omitted.

Suppose T is a fixed binary tree, and Q is a set of quartet splits defined on the leaves of T. The problem of reconstructing T from Q is in general NP-hard [46], but in [20] we showed that if $R_T \subseteq Q \subseteq Q(T)$ we can reconstruct T in $O(n^5)$ time, and validate that T is the unique tree consistent with Q. Now we define a stronger property for Q which, when it holds, will allow us to reconstruct T from Q (and validate that T is the unique tree consistent with Q) in $O(n^2 + |Q| \log |Q|)$ time. Thus, this is a faster algorithm than the DCTC algorithm that we presented in [20].

Definition 3 (*T-forcing sets of quartet splits*). A set Q of quartet splits is said to be *T-forcing* if there exists a binary tree T such that

1. $R_T \subseteq Q \subseteq Q(T)$, and
2. Q has the antiwitness property for T.

Two points should be made about this definition. Since $R_T \subseteq Q$, Q has the *witness* property for T, and it is impossible for Q to be both T-forcing and T'-forcing for distinct T and T', since by Theorem 3, R_T is consistent with a unique tree. Finally, note that the first condition $R_T \subseteq Q \subseteq Q(T)$ was the requirement we made for the dyadic closure tree construction (DCTC) algorithm in [20], and so T-forcing sets of quartet splits have to satisfy the assumptions of the DCTC algorithm, plus one additional assumption: having the *antiwitness* property.

4.5. WATC

The algorithm we will now describe operates by constructing the tree from the outside in, via a sequence of iterations. Each iteration involves determining a new set of edi-subtrees, where each edi-subtree is either an edi-subtree in the previous iteration or

is the result of making two edi-subtrees from the previous iteration siblings. Thus, each iteration involves determining which pairs of edi-subtrees from the previous iteration are siblings, and hence should be joined into one edi-subtree in this iteration.

We make the determination of siblinghood of edi-subtrees by applying the witness and antiwitness properties, but we note that only certain splits are considered to be relevant to this determination. In other words, we will require that any split used either as a witness or an anti-witness have leaves in four distinct edi-subtrees that exist at the time of the determination of siblinghood for this particular pair. Such splits are considered to be *active*, and other splits are considered to be *inactive*. All splits begin as active, but become inactive during the course of the algorithm (and once inactive, they remain inactive). We will use the terms "active witness" and "active antiwitness" to refer to active splits which are used as witnesses and antiwitesses. We will infer that two edi-subtrees are siblings if and only if there is an active witness to their siblinghood and no active anti-witness. (Note that this inference will be accurate if Q has the witness and antiwitness properties, but otherwise the algorithm may make a false inference, or fail to make any inference.)

We represent our determination of siblinghood as a graph on the edi-subtrees we have currently found. Thus, suppose at the beginning of the current iteration there are p edi-subtrees, $t_1, t_2, \ldots, t_p$. The graph for this iteration has p nodes, one for each edi-subtree, and we put an edge between every pair of edi-subtrees which have at least one witness and no anti-witness in the set of quartet topologies. The algorithm proceeds by then merging pairs of sibling edi-subtrees (recognized by edges in the graph) into a single (new) edi-subtree. The next iteration of the algorithm then requires that the graph is reconstructed, since witnesses and antiwitnesses must consist of four leaves, each drawn from distinct edi-subtrees (these are the *active* witnesses and antiwitnesses – thus, quartet splits begin as active, but can become inactive as edi-subtrees are merged).

The last iteration of the algorithm occurs when the number of edi-subtrees left is four, *or* there are no pairs of edi-subtrees which satisfy the conditions for siblinghood. If no pair of edi-subtrees satisfy the criteria for being siblings, then the algorithm returns *Fail*. On the other hand, if there are exactly four edi-subtrees, and if there are two disjoint pairs of sibling edi-subtrees, then we return the tree formed by merging each of the two pairs of sibling edi-subtrees into a single edi-subtree, and then joining the roots of these two (new) edi-subtrees by an edge.

If a tree T' is reconstructed by the algorithm, we will not return T' until we verify that

$$R_{T'} \subseteq Q \subseteq Q(T').$$

If the tree T' passes this test, then we return T', and in all other cases we return *Fail*.

We summarize this discussion in the following:

The WATC algorithm

Stage I:

- Start with every leaf of T defining an *edi*-subtree.

- While there are at least four *edi*-subtrees do:
  - Form the graph G on vertex set given by the edi-subtrees, and with edge set defined by siblinghood; i.e., $(x,y) \in E(G)$ if and only if there is at least one witness and no antiwitness to the siblinghood of edi-subtrees x and y. All witnesses and antiwitnesses must be splits on four leaves in which each leaf lies in a distinct edi-subtree; these are the *active* witnesses and antiwitnesses.
    - *Case: there are exactly four edi-subtrees*: Let the four subtrees be x, y, z, w. If the edge set of the graph G is $\{(x,y),(z,w)\}$, then construct the tree T formed by making the edi-subtrees x and y siblings, the edi-subtrees z and w siblings, and adding an edge between the roots of the two new edi-subtrees; else, return *Fail*.
    - *Case: there are more than four edi-subtrees*: If the graph has at least one edge, then select one, say (x,y), and make the roots of the *edi*-subtrees x and y children of a common root r, and replace the pair x and y by one *edi*-subtree. If no component edge exists, then Return *Fail*.

Stage II

- Verify that T satisfies the constraints $R_T \subseteq Q \subseteq Q(T)$. If so, return T, and else return *Fail*.

The runtime of this algorithm depends upon how the two *edi*-subtrees are found that can be siblings.

4.6. Implementation of WATC

We describe here a fast implementation of the WATC algorithm.

We begin by constructing a multigraph on n nodes, bijectively labelled by the species. Edges in this multigraph will be colored either green or red, with one green edge between i and j for each witness to the siblinghood of i and j, and one red edge between i and j for each antiwitness. Thus, each quartet split $ij|kl$ defines six edges in the multigraph, with two green edges $((ij)$ and $(kl))$ and four red edges $((ik),(il),(jk),(jl))$. Each green edge is annotated with the quartet that defined it and the topology on that quartet, so that the other edges associated to that quartet can be identified. Constructing this multi-graph takes $O(|Q|)$ time. Note that *edi*-subtrees x and y are determined to be siblings if there exists a green edge (x,y) but no red edge (x,y).

We will maintain several data structures:

- $Red(i,j)$, the number of red edges between nodes i and j, so that accesses, increments, and decrements to $Red(i,j)$ take $O(1)$ time,
- $Green(i,j)$, the set of green edges between nodes i and j, maintained in such a way that we can enumerate the elements in $|Green(i,j)|$ time, and so that we can union two such sets in $O(1)$ time,
- T_i, the ith edi-subtree (i.e. the edi-subtree corresponding to node i), maintained as a directed graph with edges directed away from the root,
- *Tree*, an array such that $Tree[i]=j$ indicates that leaf i is in tree T_j. This is initialized by $Tree[i]=i$ for all i, and

- *Candidates*, the set of pairs of edi-subtrees which have at least one green edge and no red edges between them (and hence are candidates for siblinghood). We maintain this set using doubly-linked lists, and we also have pointers *into* the list from other datastructures (*Green*(i,j)) so that we can access, add, and delete elements from the set in O(1) time.

Finding a sibling pair: A pair of edi-subtrees are inferred to be siblings if and only if they have at least one green edges and no red edges between them. We maintain a list of possible sibling pairs of edi-subtrees in the set *Candidates*, and the members of *Candidates* are pairs of the form i,j where both i and j are edi-subtrees. (Testing whether i is a current edi-subtree is easy; just check that $Tree[i]=i$.) We take an element (i,j) from the set *Candidates* and verify that the pair is valid. This requires verifying that both i and j are current names for *edi*-subtrees, which can be accomplished by checking that $Tree[i]=i$ and $Tree[j]=j$. If (i,j) fails this test, we delete (i,j) from the set of *Candidates*, and examine instead a different pair. However, if (i,j) passes this test, we then verify that the pair i,j have at least one green edge and no red edges between them. For technical reasons (which we describe below), it is possible that *Green*(i,j) will contain a *ghost* green edge. We now define what ghost green edges are, and how we can recognize them in O(1) time.

Definition 4. A *ghost* green edge is a green edge (a,b) which was defined by a quartet split $ab|cd$, but which was not deleted after the edi-subtrees containing c and d were merged into a single edi-subtree.

Detecting whether a green edge is a ghost is done as follows. Recall that every green edge (a,b) is annotated with the quartet (a,b,c,d) that gave rise to it. Therefore, given a green edge (a,b), we look up the edi-subtrees for the members of the *other* green edge (c,d) (using the *Tree* array), and see if c and d still belong to distinct edi-subtrees. If $Tree[c]=Tree[d]$ then (a,b) is a ghost green edge (since c and d were already placed in the same edi-subtree) and otherwise it is a true green edge.

Every ghost we find in *Green*(i,j) we simply delete, and if *Green*(i,j) contains only ghost edges, we remove (i,j) from the set *Candidates* (the edi-subtrees i and j are not actually siblings). If we find any non-ghost green edge in *Green*(i,j), then (i,j) are inferred to be sibling edi-subtrees, and we enter the next phase.

Processing a sibling pair: Having found a pair i and j of edi-subtrees which are siblings, we need to update all the data-structures appropriately. We now describe how we do this.

First, we process every green edge e in *Green*(i,j) by deleting the four red edges associated to e (this is accomplished by decrementing appropriate entries in the matrix *Red*). Note that we do not explicitly (or implicitly) delete the other green edge associated with edge e, and rather leave that green edge to be handled later; this is how *ghost* green edges arise.

After we finish processing every green edge, we merge the two edi-subtrees into one edi-subtree. We will use one index, say i, to indicate the number of the new *edi*-subtree

created. We update T_i so that it has a new root, and the children of the new root are the roots of the previous edi-subtrees T_i and T_j, and we update the *Tree* array so that all entries which previously held a j now hold i.

We also have to reset $Red(i,k)$ and $Green(i,k)$ for every other edi-subtree k, since the edi-subtree labelled i has changed. We set $Red(i,k) = Red(i,k) + Red(j,k)$, and $Green(i,k) = Green(i,k) \cup Green(j,k)$ for all k. We then set $Red(j,k) = 0$ and $Green(j,k) = \emptyset$, if we wish (this is for safety, but is not really needed).

We also have to update the *Candidates* set. This involves deletions of some pairs, and insertions of others. The only pairs which need to be deleted are those i,k for which there is now a red edge between edi-subtrees i and k, but for which previously there was none. This can be observed during the course of updating the $Red(i,k)$ entries, since every pair (i,k) which should be deleted has $Red(i,k) = 0$ before the update, and $Red(i,k) > 0$ after the update. Pairs (i,k) which must be inserted in the *Candidates* set are those (i,k) which previously had $Green(i,k) = \emptyset$ and which now have $Green(i,k) \neq \emptyset$. Accessing, inserting, and deleting the elements of *Candidates* takes O(1) time each, so this takes O(1) additional time.

We now discuss the runtime analysis of the first stage of WATC:

Theorem 5. *The first stage of WATC uses* $O(n^2 + |Q|)$ *time.*

Proof. Creating the multi-graph clearly costs only $O(|Q|)$ time. Initializing all the datastructures takes $O(n^2)$ time. There are at most $O(|Q|)$ green edges in the multigraph we create, and each green edge is processed at most once, after which it is deleted. Processing a green edge costs O(1) time, since *Tree* can be accessed in O(1) time. There are at most $n-1$ siblinghood detections, and updating the datastructures after detecting siblinghood only costs $O(n)$ time (beyond the cost of processing green edges). Implementing the datastructures $Green(i,j)$ and *Candidates* so that updates are efficient is easy through the use of pointers and records. Hence, the total cost of the first stage is $O(n^2 + |Q|)$. □

So suppose the result of the first phase constructs a tree T from the set Q of splits. The second stage of the WATC algorithm needs to verify that $R_T \subseteq Q \subseteq Q(T)$; we now describe how this is accomplished efficiently.

Given T, we can compute R_T in $O(n^2)$ time in a straightforward way: for each of the $O(n)$ edi-subtrees t, we compute the representative $rep(t)$ in $O(n)$ time. We then use the representatives to compute R_T, which has size $O(n)$, in $O(n)$ additional time. Verifying that $R_T \subseteq Q$ then takes at most $O(n \log n + |Q| \log |Q|)$ time. First we make sorted list of quartet splits by the lexicographic order of the 4 vertices involved. Sorting is in $O(|Q| \log |Q|)$ time. Then we use a binary search to determine membership, which costs $O(\log n)$ time for each element of R_T, since $|Q| = O(n^4)$. Verifying that $Q \subseteq Q(T)$ then can be done by verifying that $q \in Q(T)$ for each $q \in Q$. This is easily done in O(1) time per q using O(1) *lca* queries (to determine the valid split for each quartet which has a split in Q). Preprocessing T so that we can do *lca* queries in O(1) time

per query can be done in $O(n)$ time, using the algorithm of Harel and Tarjan [53]. Consequently, we have proven:

Theorem 6. *The second stage of WATC takes* $O(n^2 + |Q| \log |Q|)$ *time. Therefore, WATC takes* $O(n^2 + |Q| \log |Q|)$ *time.*

4.7. Proof of correctness of WATC

We begin by proving that the WATC algorithm correctly reconstructs the tree T provided that Q is T-forcing.

Theorem 7. *If Q is T-forcing, then $WATC(Q) = T$.*

Proof. We first prove that all decisions made by the algorithm are correct, and then prove that the algorithm never fails to make a correct decision.

We use induction on the number of iterations to prove that no incorrect decisions are made by the algorithm. At the first iteration, every edi-subtree is a leaf, and these are correct. Now assume that so far the WATC algorithm applied to Q has constructed only correct edi-subtrees, and the next step merges two edi-subtrees, t_1 and t_2, into one, but that these are not actually siblings.

Since Q has the antiwitness property, there is a valid quartet split $ab|cd \in Q$ with $a \in t_1, c \in t_2$ and $\{b,d\} \cap (t_1 \cup t_2) = \emptyset$. We need only show that this antiwitness is still active at the time that we merged t_1 and t_2 into one edi-subtree.

Suppose that the split $ab|cd$ is not active at the time we merged t_1 and t_2. In this case, then the four leaves a, b, c, d are in fewer than four distinct edi-subtrees. The assumption $\{b,d\} \cap (t_1 \cup t_2) = \emptyset$ then implies that we have already created an edi-subtree t containing both b and d. This edi-subtree is true, since we have assumed all edi-subtrees constructed so far are accurate. Now, consider the edge e' whose deletion creates the subtree t. This edge cannot exist if $ab|cd$ is a valid quartet split and neither b nor d are in $t_1 \cup t_2$. Consequently, the antiwitness $ab|cd$ is *still* active at the time we merged t_1 and t_2, contradicting that we made that merger, and hence all inferred edi-subtrees are correct.

We now show that the algorithm never fails to be able to make a correct decision. If Q is T-forcing, then $R_T \subseteq Q$. Now if t and t' are sibling edi-subtrees, then let e be the edge in T whose deletion disconnects $t \cup t'$ from the rest of the tree T. Let q be the representative quartet split associated to e. This quartet split is a witness to the siblinghood of t and t', which will remain active throughout the iterations of the algorithm until the entire tree is constructed (otherwise there are only three edi-subtrees present at some point, and this is contradicted by the structure of the algorithm). Furthermore, since $Q \subseteq Q(T)$, there is no invalid quartet split, and consequently no antiwitness to the siblinghood of t and t'. Therefore, the algorithm will never fail to have opportunities to merge pairs of sibling edi-subtrees. □

Theorem 8. *If the WATC algorithm returns a tree T given a set Q of quartet splits, then Q is consistent with T and with no other tree T'. If WATC does not return a tree T, then Q is not T-forcing.*

Proof. The proof is not difficult. If T is returned by WATC, then Q satisfies $R_T \subseteq Q \subseteq Q(T)$. Under this condition Q is consistent with T and with no other tree, by Theorem 3. Hence the first assertion holds. For the second assertion, if Q is T-forcing, then by the previous theorem WATC returns T after the first stage. The conditions for being T-forcing include that $R_T \subseteq Q \subseteq Q(T)$, so that the verification step is successful, and Q is returned. □

5. The witness–antiwitness method (WAM)

In the previous section we described the WATC algorithm which reconstructs T given a T-forcing set of quartet splits, Q. In this section we describe a set of search strategies for finding such a set Q. These strategies vary in their number of queries on quartet split sets (ranging from $O(\log\log n)$ to $O(n^2)$), but also vary in the sequence length needed in order for the search strategy to be successful with high probability. All have the same asymptotic sequence length requirement as the dyadic closure method [20], but differ in terms of the multiplicative constant.

Before we describe and analyze these search strategies, we begin with some results on the four-point Method, and on random trees.

5.1. Previous results

Lemma 3 (Azuma–Hoeffding inequality, see [3]). *Suppose $X = (X_1, X_2, \ldots, X_k)$ are independent random variables taking values in any set S, and $L : S^k \to \mathbb{R}$ is any function that satisfies the condition: $|L(\boldsymbol{u}) - L(\boldsymbol{v})| \leqslant t$ whenever $\boldsymbol{u}$ and $\boldsymbol{v}$ differ at just one coordinate. Then, for any $\lambda > 0$, we have*

$$\mathbb{P}[L(\boldsymbol{X}) - \mathbb{E}[L(\boldsymbol{X})] \geqslant \lambda] \leqslant \exp\left(-\frac{\lambda^2}{2t^2k}\right),$$

$$\mathbb{P}[L(\boldsymbol{X}) - \mathbb{E}[L(\boldsymbol{X}] \leqslant -\lambda] \leqslant \exp\left(-\frac{\lambda^2}{2t^2k}\right).$$

In [20], we proved:

Theorem 9. *Assume that z is a lower bound for the transition probability of any edge of a tree T in the Neyman 2-state model, $y \geqslant \max E^{ij}$ is an upper bound on the compound changing probability over all ij paths in a quartet q of T. The probability that FPM fails to return the correct quartet split on q is at most*

$$18\exp(-(1-\sqrt{1-2z})^2(1-2y)^2k/8)). \tag{8}$$

In [20] we also provided an upper bound on the growth of the depth of random trees under two distributions:

Theorem 10. (i) *For a random semilabelled binary tree T with n leaves under the uniform model, $depth(T) \leqslant (2 + o(1)) \log_2 \log_2 (2n)$ with probability $1 - o(1)$.*

(ii) *For a random semilabelled binary tree T with n leaves under the Yule-Harding distribution, after suppressing the root, $depth(T) = (1 + o(1)) \log_2 \log_2 n$ with probability $1 - o(1)$.*

5.2. Search strategies

Let Q_w denote the set of splits inferred using the four-point method on quartets whose width is at most w; recall that the width of a quartet i,j,k,l is the maximum of $d_{ij}, d_{ik}, d_{il}, d_{jk}, d_{jl}, d_{kl}$. The objective is to find a set Q_w such that Q_w is T-forcing.

Definition 5.

$$\mathscr{A} = \{w \in \mathbb{R}^+ \colon R_T \subseteq Q_w\},$$

$$\mathscr{B} = \{w \in \mathbb{R}^+ \colon Q_w \subseteq Q(T)\}.$$

We now state without proof the following observation which is straightforward.

Observation 1. *$\mathscr{A}$ is either $\emptyset$, or is (w_A, ∞) for some positive real number w_A. $\mathscr{B}$ is either $\emptyset$, or is $(0, w_B)$, for some positive real number w_B.*

Sequential search for T-forcing Q_w: A sequential search through the sets Q_w, testing each Q_w for being T-forcing by a simple application of WATC algorithm, is an obvious solution to the problem of finding a T-forcing set which will find a T-forcing set from shorter sequences than any other search strategy through the sets Q_w. However, in the worst case, it examines $O(n^2)$ sets Q_w, since w can be any of the values in $\{d_{ij} \colon 1 \leqslant i < j \leqslant n\}$, and hence it has high computational requirements.

Sparse-high search for a T-forcing Q_w: We describe here a sparse search that examines at most $O(\log k)$ sets Q_w and hence has lower computational requirements, but may require longer sequences. Even so, we prove that the sequence length requirement has the same order of magnitude as the sequential search. This sparse search examines the high end of the values of w, and so we call it the *Sparse-high* search strategy.

Let $\tau < 1/4$ be given. We define Z_τ to be the set of quartets i,j,k,l such that $\max\{h^{ij}, h^{ik}, h^{il}, h^{jk}, h^{jl}, h^{kl}\} < 1/2 - 2\tau$. Note then that the set of splits (inferred using the four-point method) on quartets in Z_τ is $Q_{w(\tau)}$, where $w(\tau) = -\frac{1}{2}(\log(4\tau))$.

The sparse-high search examines $\tau = 1/8, 1/16, \ldots$, until it finds a τ such that $Z_\tau = Q_{w(\tau)}$ is T-forcing, or until $w(\tau)$ exceeds every d_{ij}.

We now define conditions under which each of these search strategies are guaranteed to find a T-forcing set Q_w. Recall the sets $\mathscr{A} = \{w \colon R_T \subseteq Q_w\}$,and $\mathscr{B} = \{w \colon Q_w \subseteq Q(T)\}$.

We now define the following *assumptions*:

$$\mathscr{A} \cap \mathscr{B} \neq \emptyset, \tag{9}$$

$$\exists w^* \in \mathscr{A} \cap \mathscr{B}, \text{ s.t. } Q_{w^*} \text{ has the antiwitness property,} \tag{10}$$

$$\exists \tau^*, \text{ s.t. } \forall \tau \in [\tau^*/2, \tau^*], w(\tau) \in \mathscr{A} \cap \mathscr{B}, \text{ and } Q_{w(\tau)} \text{ has the antiwitness property.} \tag{11}$$

It is clear that if assumptions (16) and (17) hold, then the sequential search strategy will be guaranteed to succeed in reconstructing the tree, and that the Sparse-high search strategy requires that assumption (11) hold as well.

We now analyze the sequence length needed to get each of these assumptions to hold with constant probability.

6. How WAM performs under the Neyman 2-state model

In this section we analyze the performance of the witness–antiwitness method (WAM), with respect to computational and sequence-length requirements. The analysis of the sequence length requirement follows a similar analysis for DCM in [20], but turns out to be more complicated, and results in constant times longer sequences. The analysis of the computational complexity of WAM is both in the worst case, and under the assumption that the tree topology is drawn from a random distribution. Finally, we compare the performance of WAM to other methods, with respect to both these issues.

6.1. Sequence length needed by WAM

Theorem 11. *Suppose k sites evolve under the Cavender–Farris model on a binary tree T, so that for all edges e, $p_e \in [f, g]$, where we allow $f = f(n)$ and $g = g(n)$ to be functions of n. We assume that $\limsup_n g(n) < 1/2$. Then both the sparse-high and sequential search based on the WATC algorithm returns the true tree T with probability $1 - o(1)$, if*

$$k > \frac{c \cdot \log n}{(1 - \sqrt{1-2f})^2 (1-2g)^{4\text{depth}(T)}}, \tag{12}$$

where c is a fixed constant.

Proof. Note that the sparse-high search requires assumptions (16)–(18), while the sequential search only requires assumptions (16) and (17). We will show that the given sequence length suffices for all three assumptions to hold with probability $1 - o(1)$.

We begin by showing that assumption (9) holds, i.e. that $R_T \subseteq Q_w \subseteq Q(T)$ for some w.

For k evolving sites (i.e. sequences of length k), and fixed $\tau > 0$, let us define the following two sets:

$$S_\tau = \{\{i,j\}: h^{ij} < 0.5 - \tau\},$$

and

$$Z_\tau = \left\{ q \in \binom{[n]}{4} : \text{for all } i,j \in q, \{i,j\} \in S_{2\tau} \right\},$$

and the following four events:

$$A = Q_{\text{short}}(T) \subseteq Z_\tau, \tag{13}$$

$$B_q = \text{FPM correctly returns the split of the quartet } q \in \binom{[n]}{4}, \tag{14}$$

$$B = \bigcap_{q \in Z_\tau} B_q, \tag{15}$$

$$C = S_{2\tau} \text{ contains all } \{i,j\} \text{ with } E^{ij} < 0.5 - 3\tau \text{ and no } \{i,j\} \text{ with } E^{ij} \geqslant 0.5 - \tau. \tag{16}$$

Note that B is the event that $Q_{w(\tau)} \subseteq Q(T)$, so that $A \cap B$ is the event that $Q^*_{\text{short}} \subseteq Q_{w(\tau)} \subseteq Q(T)$, or $w(\tau) \in \mathscr{A} \cap \mathscr{B}$. Thus, $\mathbb{P}[\mathscr{A} \cap \mathscr{B} \neq \emptyset] \geqslant \mathbb{P}[A \cap B]$. Define

$$\lambda = (1 - 2g)^{2\text{depth}(T)+3}. \tag{17}$$

We claim that

$$\mathbb{P}[C] \geqslant 1 - (n^2 - n)\mathrm{e}^{-\tau^2 k/2} \tag{18}$$

and

$$\mathbb{P}[A|C] = 1 \quad \text{if } \tau \leqslant \lambda/6. \tag{19}$$

To establish (18), first note that h^{ij} satisfies the hypothesis of the Azuma–Hoeffding inequality (Lemma 3 with $X_l = 1$ if the lth bits of the sequences of leaves i and j differ, and $X_l = 0$ otherwise, and $t = 1/k$). Suppose $E^{ij} \geqslant 0.5 - \tau$. Then,

$$\begin{aligned}\mathbb{P}[\{i,j\} \in S_{2\tau}] &= \mathbb{P}[h^{ij} < 0.5 - 2\tau] \\ &\leqslant \mathbb{P}[h^{ij} - E^{ij} \leqslant 0.5 - 2\tau - E^{ij}] \leqslant \mathbb{P}[h^{ij} - \mathbb{E}[h^{ij}] \leqslant -\tau] \leqslant \mathrm{e}^{-\tau^2 k/2}.\end{aligned}$$

Since there are at most $\binom{n}{2}$ pairs $\{i,j\}$, the probability that at least one pair $\{i,j\}$ with $E^{ij} \geqslant 0.5 - \tau$ lies in $S_{2\tau}$ is at most $\binom{n}{2}\mathrm{e}^{-\tau^2 k/2}$. By a similar argument, the probability that $S_{2\tau}$ fails to contain a pair $\{i,j\}$ with $E^{ij} < 0.5 - 3\tau$ is also at most $\binom{n}{2}\mathrm{e}^{-\tau^2 k/2}$. These two bounds establish (18).

We now establish (19). For $q \in Q_{\text{short}}(T)$ and $i,j \in q$, if a path $e_1 e_2 \cdots e_t$ joins leaves i and j, then $t \leqslant 2depth(T) + 3$ by the definixtion of $Q_{\text{short}}(T)$. Using these facts,

Lemma 1, and the bound $p_e \leqslant g$, we obtain $E^{ij} = 0.5[1-(1-2p_1)\cdots(1-2p_t)] \leqslant 0.5(1-\lambda)$. Consequently, $E^{ij} < 0.5-3\tau$ (by assumption that $\tau \leqslant \lambda/6$) and so $\{i,j\} \in S_{2\tau}$ once we condition on the occurrence of event C. This holds for all $i,j \in q$, so by definition of Z_τ we have $q \in Z_\tau$. This establishes (19).

Define a set

$$X = \left\{ q \in \binom{[n]}{4} : \max\{E^{ij} : i,j \in q\} < 0.5 - \tau \right\}$$

(note that X is not a random variable, while Z_τ, S_τ are). Now, for $q \in X$, the induced subtree in T has mutation probability at least $f(n)$ on its central edge, and mutation probability of no more than $\max\{E^{ij} : i,j \in q\} < 0.5-\tau$ on any pendant edge. Then, by Theorem 9 we have

$$\mathbb{P}[B_q] \geqslant 1 - 18\exp(-(1-\sqrt{1-2f})^2\tau^2 k/8) \tag{20}$$

whenever $q \in X$. Also, the occurrence of event C implies that

$$Z_\tau \subseteq X \tag{21}$$

since if $q \in Z_\tau$, and $i,j \in q$, then $i,j \in S_{2\tau}$, and then (by event C), $E^{ij} < 0.5-\tau$, hence $q \in X$. Thus,

$$\mathbb{P}[B \cap C] = \mathbb{P}\left[\left(\bigcap_{q \in Z_\tau} B_q\right) \cap C\right] \geqslant \mathbb{P}\left[\left(\bigcap_{q \in X} B_q\right) \cap C\right],$$

where the second inequality follows from (21), as this shows that when C occurs, $\bigcap_{q \in Z_\tau} B_q \supseteq \bigcap_{q \in X} B_q$. Invoking the Bonferonni inequality, we deduce that

$$\mathbb{P}[B \cap C] \geqslant 1 - \sum_{q \in X} \mathbb{P}[\overline{B_q}] - \mathbb{P}[\overline{C}]. \tag{22}$$

Thus, from above,

$$\mathbb{P}[A \cap B] \geqslant \mathbb{P}[A \cap B \cap C] = P[B \cap C]$$

(since $\mathbb{P}[A|C] = 1$), and so, by (20) and (22),

$$\mathbb{P}[A \cap B] \geqslant 1 - 18\binom{n}{4}\exp(-(1-\sqrt{1-2f})^2\tau^2 k/8) - (n^2-n)e^{-\tau^2 k/2}.$$

Formula (12) follows by an easy calculation for $\tau = c \cdot \lambda$, for any $0 < c \leqslant 1/6$.

We proceed to prove that assumption (10) holds. Recall the definition of $Q_{w(\tau)} = \{FPM(q) : q \in Z_\tau\}$. Now let D be the event that whenever t and t' are two edi-subtrees which are *not* siblings, but there is a witness in $Q_w(\tau)$ to the siblinghood of T, then there is also an antiwitness in $Q_w(\tau)$.

Recalling Theorem 4, it is obvious that event $A \cap B \cap D$ implies Assumptions (9) and (10). We are going to show that $\mathbb{P}[A \cap B \cap D] = 1 - o(1)$ under the conditions of

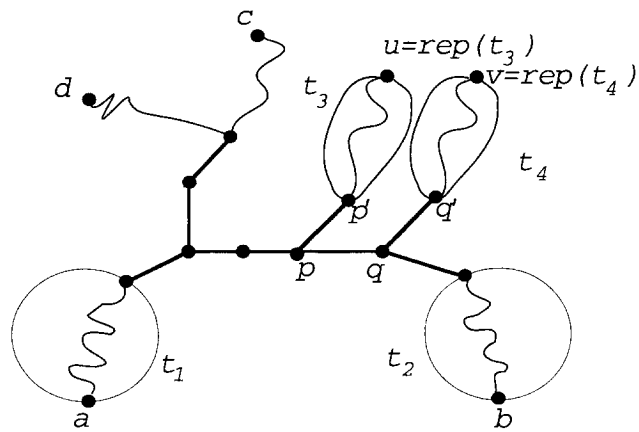


Fig. 1. Finding an antiwitness.

the theorem for a certain choice of τ, which is just slightly smaller than the τ that sufficed for the assumption (9). Technically, we are going to show

$$\mathbb{P}[D|A\cap B\cap C]=1. \tag{23}$$

proof of (23): $\overline{D}=\bigcup_{t_1,t_2} H_{t_1,t_2}$, where t_1, t_2 denote two disjoint edi-subtrees of T, and H_{t_1,t_2} denotes the event that there is a witness but no antiwitness for the siblinghood of t_1, t_2 in $Q_{w(\tau)}$. Therefore, in order to prove (23), it suffices to prove

$$\mathbb{P}[H_{t_1,t_2}|A\cap B\cap C]=0. \tag{24}$$

Assume that there is a witness for the siblinghood of t_1, t_2 where t_1 and t_2 are *not* siblings. We will show that $Q_{w(\tau)}$ contains an antiwitness to the siblinghood of t_1 and t_2. Let the witness to the siblinghood of t_1 and t_2 be $ab|cd$, where $a\in t_1$, $b\in t_2$, and c,d not in $t_1\cup t_2$. Let pq be an internal edge of the unique ab path in T containing the midpoint of the path $P(a,b)$ measured using the lengths defined by the corrected model distances D, and with p closer to a and q closer to b, i.e. the edge (p,q) maximizes the following quantity:

$$\min_{pq\ internal\ edge}(1-2E^{qp}, 1-2E^{qb}). \tag{25}$$

Let p' and q' be neighbors of p and q respectively that are not on the path between nodes a and b. Consider the *edi*-subtrees t_3 and t_4 rooted at p' and q' respectively, formed by deleting (p,p') and (q,q'), respectively. Set $u=rep(t_3)$, $v=rep(t_4)$ (Fig. 1).

We are going to show that

$$\{a,b,u,v\}\in Z_\tau, \tag{26}$$

and $au|bv\in Q_{w(\tau)}$. The proof of (26) is the only issue, since by (15) the split of $\{a,b,u,v\}$ is correctly reconstructed, and is $au|bv$ by construction. Clearly

$$\mathbb{P}[H_{t_1,t_2}|A\cap B\cap C]\leqslant\mathbb{P}[\{a,b,u,v\}\notin Z_\tau|A\cap B\cap C]. \tag{27}$$

The RHS of (27) can be further estimated by

$$\begin{aligned}&\mathbb{P}[h^{au} \geqslant 0.5 - 2\tau | A \cap B \cap C] + \mathbb{P}[h^{av} \geqslant 0.5 - 2\tau | A \cap B \cap C] \\ &\mathbb{P}[h^{bu} \geqslant 0.5 - 2\tau | A \cap B \cap C] + \mathbb{P}[h^{bv} \geqslant 0.5 - 2\tau | A \cap B \cap C] \\ &\qquad + \mathbb{P}[h^{uv} \geqslant 0.5 - 2\tau | A \cap B \cap C]. \end{aligned} \tag{28}$$

The fifth term $\mathbb{P}[h^{uv} \geqslant 0.5 - 2\tau | A \cap B \cap C] = 0$, since it is easy to find a short quartet which contains u, v; and therefore by (13), $h^{uv} < 0.5 - 2\tau$. Here is how to find a short quartet containing u and v. Let a' denote the neighbor of p on the ab path towards a, and let q denote the neighbor of q on the ab path towards b. Consider the edi-subtree t_5 defined by pa', which contains the leaf a, and the edi-subtree t_6 defined by qb', which contains the leaf b. It is easy to check that $\{u, v, rep(t_5), rep(t_6)\}$ is a short quartet.

In order to finish the proof of (24), and hence the proof of (23), it suffices to show that the other four terms in (28) are zero as well. The third and fourth terms are symmetric to the first and second, and in fact the second has a worse bound than the first. Therefore it suffices to prove that

$$\mathbb{P}[h^{av} \geqslant 0.5 - 2\tau | A \cap B \cap C] = 0. \tag{29}$$

We assume that $\{a, v\} \notin S_{2\tau}$, and show that consequently τ is large. Hence, for a properly small τ, Formula (29), and hence (23) holds. From $\{a, v\} \notin S_{2\tau}$, conditioning on C,

$$E^{av} > 0.5 - 3\tau, \tag{30}$$

and $\{a, b\} \in S_{2\tau}$, and hence, conditioning on C,

$$E^{ab} < 0.5 - \tau. \tag{31}$$

There is no difficulty to extend the definition of E^{ij} to cases when at least one of i, j is an internal vertex of the tree. Simple algebra yields from formula (30) and Lemma 1, that

$$6\tau \geqslant 1 - 2E^{av} = (1 - 2E^{pv})(1 - 2E^{pa}). \tag{32}$$

We have

$$1 - 2E^{pv} \geqslant (1 - 2g)^{depth(T)+2} = \sqrt{\lambda(1 - 2g)} \tag{33}$$

by the definition of λ (see formula (17)) and the choice of v as representative. By formula (25), it is easy to see that

$$1 - 2E^{pa} \geqslant q(1 - 2g)^2 \sqrt{1 - 2E^{ab}}. \tag{34}$$

Combining (31)–(34), we obtain $6\tau > \sqrt{\lambda(1 - 2g)}(1 - 2g)^2 \sqrt{2\tau}$. This formula fails, if we select

$$\tau = c_2 \cdot (1 - 2g)^5 \lambda \tag{35}$$

with a sufficiently small positive constant c_2.

Case 1: $p \notin t_1$ *and* $q \notin t_2$ (*as in Fig.* 1). Then $au|bv \in Q_{w(\tau)}$ is an anti-witness, as desired.

When Case 1 does not hold, the only problem that can arise is if the valid split $au|bv$ does not satisfies the condition $\{u,v\} \cap (t_1 \cup t_2) = \emptyset$, and hence is not an antiwitness.

Case 2: $p \in t_1$ *or* $q \in t_2$. Without loss of generality we may assume $p \in t_1$. Now we *redefine* the location of the edge pq on the ab path as follows. Let p denote the first vertex after $root(t_1)$ on the ab path and let q denote the second. Clearly $q \notin t_2$, since t_1 and t_2 are not siblings. We also redefine p', q', t_3, u, t_4, v according to the new p and q. Redefine a to be $rep(t_1)$ and call the old a as a^*. Now we are going to show (26) and that $au|bv \in Q_{w(\tau)}$ is the sought-for antiwitness (note a, u, v have been redefined, but b has not). Again, we have to see (27) and prove that (28) is termwise zero.

For pairs u, v where $\{u,v\} \in S_{2\tau}$, we proceed exactly as in Case 1. Observe that E^{bu} and E^{bv} *decreased* during the redefinition, so a calculation like (29)–(35) still goes through. Observe that $L(a,u) \leqslant 2depth(T)+2$, $L(a,v) \leqslant 2depth(T)+3$, and hence $\{a,u\} \in S_{2\tau}$ and $\{a,v\} \in S_{2\tau}$, exactly as in the proof of (19). The only thing left to prove is $\{a,b\} \in S_{2\tau}$.

In order to prove $\mathbb{P}[h^{ab} \geqslant 0.5 - 2\tau | A \cap B \cap C] = 0$, since under the condition C, it suffices to prove $1 - 2E^{ab} > 6\tau$. However,

$$1 - 2E^{ab} = (1 - 2E^{a,root(t_1)})(1 - 2E^{root(t_1),b}) \geqslant (1-2g)^{depth(T)}(1-2g)^2\sqrt{1 - E^{a^*b}},$$

and we still have $\sqrt{1 - E^{a^*b}} > \sqrt{2\tau}$ according to (31). A calculation like the one resulting in (35) gives the result wanted, and we are finished with the proof of (23).

Using these statements, $\mathbb{P}[A \cap B \cap D] \geqslant \mathbb{P}[A \cap B \cap D | A \cap B \cap C] \times \mathbb{P}[A \cap B \cap C] = \mathbb{P}[A \cap B \cap C] = \mathbb{P}[B \cap C]$, and we are back to the same estimates that proved assumption (9), but we need a slightly smaller τ and consequently slightly larger k.

Note that the proof above applies to all $c_3 \in [c_2/2, c_2]$, if it applies to $c_3 = c_2$ and $c_3 = c_2/2$, so that assumption (11) holds. □

Note that the proof also handled the problem that arises if some of the dissimilarity scores exceed 1/2, and so we cannot even compute corrected distances. The moral is that those pairs are not needed according to the proof. Therefore there is no need for additional conditioning for the shape of the observed data.

6.2. *Runtime analysis of the search strategies*

Theorem 12. (i) *The running time of WAM based on sequential search is* $O(n^2k + n^6 \log n)$

(ii) *The running time of WAM based on sparse-high search is* $O(n^2k + n^4 \log n \log k)$. *Assume now that our model tree is a random binary tree, under the uniform or Yule–Harding distribution, and all mutation probabilities are taken from an interval* $(p - \varepsilon_n, p + \varepsilon_n)$, *for a sufficiently small sequence* ε_n. *If* k *is as large as in* (12), *then with probability* $1 - o(1)$

(iii) *The running time of WAM based on sequential search is* $O(n^2k + n^3 poly \log n)$.

(iv) *The running time of WAM based on sparse-high search is* $O(n^2k + n^2 poly \log n)$.

Proof. Computing the matrices h and d takes $O(n^2k)$ time. (All distance methods begin by computing these distance matrices, but this "overhead cost" is usually always mentioned in the running time analysis of a given method.) Let w_0 be defined to be the smallest $w \in h^{ij}$ such that Q_w is T-forcing. Let $i(w)$ be the *order* of w within the sorted h^{ij} values. Then, since each call of the WATC algorithm uses $O(n^2 + |Q| \log |Q|)$ time, the running time of the sequential search is $O(i(w_0)(n^2 + |Q_{w_0}| \log |Q_{w_0}|))$, after the preprocessing.

For (i), the sequential search application of the WATC algorithm is $O(n^6 \log n)$, since we need never do more than examine all sets Q_w, and the largest such set has cardinality $O(n^4)$.

Claim (ii) follows form the observations that the sparse-high search calls the WATC algorithm at most $O(\log k)$ times, and each call costs at most $O(n^4 \log n)$ time.

We now prove (iii). The depth of a random tree (under either the uniform or Yule–Harding distributions) is with high probability $O(\log \log n)$ by Theorem 10, and so there are at most $O(poly \log n)$ leaves which are no more than about $O(\log \log n)$ distance (measured topologically) from any fixed leaf. This is the only fact that we exploit from the assumption of randomness of the tree. For two leaves i, j, recall that $L(i, j)$ denotes the topological distance between i and j. We are going to show that if τ is the value at which the search reconstructs the tree in the proof of Theorem 11, then with probability $1 - o(1)$ we have $L(i, j) = O(\log \log n)$, whenever $i, j \in q \in Q_\tau$. This yields $|Q_{w(\tau)}| = n \cdot poly \log(n)$. In the proof of Theorem 11, according to formula (18), event C holds with probability $1 - o(1)$. In that proof $Q_{w(\tau)}$ is denoted by $Z_{\tau/4}$. Now

$$(1 - 2g)^{L(i,j)} = 1 - 2E^{ij} \geqslant \tau/2, \tag{36}$$

where the equality follows from Lemma 1, and the inequality follows from the conditioning on the event C. Plugging in (35) for τ immediately yields $L(i, j) = O(\log \log n)$. Since the sequential search makes $O(npoly \log(n))$ calls to the WATC algorithm, (iii) follows.

To obtain (iv), observe that Formulae (35), (17), and $depth(T) = O(\log \log n)$ imply that the number of iterations in the sparse-high search is

$$-\log_2 \tau = O(-\log(1 - 2g) \cdot depth(T)) = O(\log \log n). \qquad \square$$

6.3. *The performance of other distance methods under the Neyman 2-state model*

In this section we describe the convergence rate for the WAM and DCM method, and compare it briefly to the rates for two other distance-based methods, the Agarwala et al. 3-approximation algorithm [1] for the L_∞-nearest tree, and neighbor joining [43]. We make the natural assumption that all methods use the same corrected empirical distances from Neyman 2-state model trees. The comparison we provide in this section will establish that our method requires *exponentially shorter* sequences in order to ensure accuracy of the topology estimation than the algorithm of Agarwala et al., for almost all trees under uniform or Yule–Harding probability distributions. The trees for which the two methods need comparable sequence lengths are those in which the

diameter and the depth are as close as possible – such as complete binary trees. Even in these cases, WAM and DCM will nevertheless need shorter sequences than Agarwala et al. to obtain the topology with high probability, as we showed it in Section 3. (Again, note that this analysis is inherently pessimistic, and it is possible that the methods may obtain accurate reconstructions from shorter sequences than suffice by this analysis.)

The neighbor joining method is perhaps the most popular distance-based method used in phylogenetic reconstruction, and in many simulation studies (see [34, 35, 44] for an entry into this literature) it seems to outperform other popular distance based methods. The Agarwala et al. algorithm [1] is a distance-based method which provides a 3-approximation to the L_∞ nearest tree problem, so that it is one of the few methods which provide a provable performance guarantee with respect to any relevant optimization criterion. Thus, these two methods are two of the most promising distance-based methods against which to compare our method. All these methods use polynomial time.

In [22], Farach and Kannan analyzed the performance of the Agarwala et al. algorithm with respect to tree reconstruction in the Neyman 2-state model, and proved that the Agarwala et al. algorithm converged quickly for the variational distance. Personal communication from S. Kannan gave a counterpart to (12): if T is a Neyman 2-state model tree with mutation rates in the range $[f, g]$, and if sequences of length k' are generated on this tree, where

$$k' > \frac{c' \cdot \log n}{f^2(1-2g)^{2diam(T)}} \tag{37}$$

for an appropriate constant c', and where $diam(T)$ denotes the "diameter" of T, then with probability $1 - o(1)$ the result of applying Agarwala et al. to corrected distances will return the topology of the model tree. In [5], Atteson proved the same result for Neighbor Joining though with a different constant. (The constant for neighbor joining is smaller than the constant for the Agarwala et al. algorithm, suggesting that neighbor joining can be guaranteed to be accurate from shorter sequences than Agarwala et al., on any tree in the Neyman 2-state model. However, remember that this analysis is pessimistic, and it may be that correct reconstruction is possible from shorter sequences than this analysis suggests.)

Comparing this formula to (12), we note that the comparison of depth and diameter is the issue, since $(1 - \sqrt{1-2f})^2 = \Theta(f^2)$ for small f. It is easy to see that $diam(T) \geqslant 2depth(T)$ for binary trees T, but the diameter of a tree can in fact be quite large (up to $n-1$), while the depth is never more than $\log n$. Thus, for every fixed range of mutation probabilities, the sequence length that suffices to guarantee accuracy for the Neighbor Joining or Agarwala et al. algorithms can be quite large (i.e. it can grow exponentially in the number of leaves), while the sequence length that suffices for the witness-antiwitness method will never grow more than polynomially.

In order to understand the bound on the sequence length needed by these methods, we now turn to an analysis of the diameter of random trees. The models for random trees are the *uniform* model, in which each tree has the same probability, and the

Yule–Harding model, studied in [2, 8, 29]. This distribution is based upon a simple model of speciation, and results in "bushier" trees than the uniform model.

Theorem 13. (i) *For a random semilabelled binary tree T with n leaves under the uniform model, $diam(T) > \varepsilon\sqrt{n}$ with probability $1 - O(\varepsilon^2)$.*

(ii) *For a random semilabelled binary tree T with n leaves under the Yule–Harding distribution, after suppressing the root, $diam(T) = \Theta(\log n)$, with probability $1 - o(1)$.*

Proof. We begin by establishing (i). The result of Carter et al. [11] immediately implies that leaves a, b have distance $m+1$ with probability *exactly* $m!N(n-2,m)/(2n-5)!!$ under the uniform model. For small enough ε, $m \leqslant \varepsilon\sqrt{n}$, this probability is $\Theta(m/n)$. Summing up the probabilities from $m = 1$ to $m = \varepsilon\sqrt{n}$, we see that $diam(T) > \varepsilon\sqrt{n}$ with probability at least $1 - O(\varepsilon^2)$.

We now consider (ii). First we describe rooted Yule–Harding trees. These trees are defined by the following constructive procedure. Make a random permutation $\pi_1, \pi_2, \ldots, \pi_n$ of the n leaves, and join π_1 and π_2 by edges to a root R of degree 2. Add each of the remaining leaves sequentially, by randomly (with the uniform probability) selecting an edge incident to a leaf in the tree already constructed, subdividing the edge, and make π_i adjacent to the newly introduced node. For a rooted Yule–Harding tree T^R, let $h(T^R)$ denote the maximum distance of any leaf from the root. Let T be the unrooted Yule–Harding tree obtained from T^R by suppressing the root, and identifying the two edges incident with the root. Let $diam(T)$ denote the diameter of T. Then, we always have

$$h(T^R) \leqslant diam(T) \leqslant 2h(T^R) - 1.$$

Now Aldous [2] shows that $h(T^R)/\log n$ converges in distribution to a (nonzero) constant c. Then, with probability tending to 1, $diam(T)/\log n$ will lie between c and $2c$. □

In Table 1, we summarize sequence length that *suffice* for accurate reconstruction with high probability of WAM and DCM, and compare these to the sequence lengths that suffice for the Agarwala et al. algorithm, according to the analyses that we have given above (thus, our summary is based upon (12), (37), and Theorems 10 and 13). Sequence lengths are given in terms of growth as a function of n, and assume that mutation probabilities on edges lie within the specified ranges.

7. Extension to general stochastic models

In this section we consider the generalization of the WAM and DCM for inferring trees in the general stochastic model. Just as in the case of the Neyman 2-state model, we find that WAM and DCM obtains accurate estimations of the tree from sequences whose length is never more than polynomial in the number of leaves (for every fixed

Table 1

		Range of mutation probabilities on edges	
		$[f,g]$ f,g are constants	$\left[\frac{1}{\log n}, \frac{\log\log n}{\log n}\right]$
Binary trees	DCM/WAM	Polynomial	Polylog
Worst-case	Agarwala et al.	Superpolynomial	Superpolynomial
Random binary trees	DCM/WAM	Polylog	Polylog
(uniform model)	Agarwala et al.	Superpolynomial	Superpolynomial
Random binary trees	DCM/WAM	Polylog	Polylog
(Yule–Harding)	Agarwala et al.	Polynomial	Polylog

range for the mutation probabilities), and in general only polylogarithmic in the number of leaves. This should be contrasted to the study of Ambainis et al. [4].

Suppose the sequence sites evolve *i.i.d.* according to the "general" Markov model – that is, there is some distribution of states π at the root of the tree, and each edge e has an associated stochastic transition matrix $M(e)$, and the (random) state at the root evolves down the tree under a natural Markov assumption, as in the general stochastic model of Definition (III).

Let $f_{ij}(\alpha,\beta)$ denote the probability that leaf i is in state α and leaf j is in state β. By indexing the states, $f_{ij}(\alpha,\beta)$ forms a square matrix, $F_{ij}=[f_{ij}(\alpha,\beta)]$. Then

$$\phi_{ij} = -\log\det(F_{ij}) \tag{38}$$

denotes the *corrected model distance* between i and j. (There will be a guarantee for $\det(F_{ij})>0$.)

The *corrected empirical distance* $\hat{\phi}_{ij}$ of two species is computed as in (38), but uses the matrix $\hat{F}_{ij}$ composed of the relative frequencies $\hat{f}_{ij}(\alpha,\beta)$ of i being in state α and j being in state β, instead of the probability $f_{ij}(\alpha,\beta)$:

$$\hat{\phi}_{ij} = -\log\det(\hat{F}_{ij}). \tag{39}$$

Then, ϕ_{ij} can be derived from a positive edge weighting of the model tree, provided that the identifiability condition described in Section 2 (Tree Reconstruction) holds. These mild conditions only require that $\det(M(e))$ not take on the values $0, 1, -1$, and that the components of π are nonzero (i.e. every state has a positive probability of occurrence at the root).

Note that $\det(M(e))$ takes the values 1 or -1 precisely if $M(e)$ is a permutation matrix. Also, for the Neyman 2-state model $\det(M(e))=1-2p(e)$, where $p(e)$ is the mutation probability on edge e; thus, $\det(M(e))>0$ and $\det(M(e))$ tend to 0 as p approaches 0.5, and tend to 1 as p approaches 0. In general, $(1/2)[1-\det(M(e)]$ plays the role of $p(e)$ in the general model. Thus, a natural extension of our restriction $f\leqslant p(e)\leqslant g$ and from the Neyman 2-state model corresponds to

$$0<1-2x'\leqslant\det(M(e))\leqslant 1-2x<1, \tag{40}$$

for suitable x, x', and we will henceforth impose this restriction for all edges of the tree. For technical reasons, we also impose the mildly restrictive condition that every vertex can be in each state μ with at least a certain fixed positive probability:

$$\pi(v)_\mu > \varepsilon. \tag{41}$$

This condition (41) certainly holds under the Neyman 2-state model, the Kimura 3-state model [39], and much more general models (providing each state has positive probability of occurring at the root). Indeed this last weaker condition might be enough, but it would seem to complicate the analysis quite a lot.

Now, let $\lambda(e)$ be the weight of edge e in the realization of ϕ on the (unrooted version) of the true underlying tree T.

Lemma 4. *Set* $\delta(x) = -0.5\log(1-2x)$. *Then*

$$\lambda(e) \geqslant -0.5\log(\det(M(e))) \geqslant \delta(x) \tag{42}$$

for every edge e of T.

Proof. The second inequality follows from the restriction we imposed above on $\det(M(e))$. The first inequality in (42) follows from similar arguments to those appearing in [47]; for the sake of completeness we give a proof.

Let T be the unrooted version of T^ρ. Now the edges of T correspond bijectively to the edges of T^ρ, except perhaps for one *troublesome* edge of T which arises whenever the root of T^ρ has degree two – in that case, two edges e_1, e_2 of T^ρ adjacent to ρ are identified to form e. For convenience, we assume in this proof that ρ is not a leaf.

We now prove that $\lambda(e) \geqslant -0.5\log\det(M(e))$ for all (non-troublesome) edges e of T, and if T has a troublesome edge e corresponding to edges e_1 and e_2 in T^ρ, then $\lambda(e) \geqslant -0.5\log(\det(M(e_1))\det(M(e_2)))$.

For any edge $e=(v,w)$ of T^ρ where w is a leaf, let

$$h(e) = -\log\det(M(e)) - 0.5\log\left[\prod_\mu \pi(v)_\mu\right]$$

while, for any edge $e=(v,w)$ of T^ρ for which neither of v, w are leaves, let

$$h(e) = -\log\det(M(e)) - 0.5\log\left[\prod_\mu \pi(v)_\mu\right] + 0.5\log\left[\prod_\mu \pi(w)_\mu\right].$$

Thus, h describes a weighting of the edges of T^ρ and thereby a weighting h^* of the edges of T by setting h^* equal to h on the non-troublesome edges, and the convention that if T has a troublesome edge e arising from the identification of a pair e_1, e_2 of edges of T^ρ then $h^*(e) = h(e_1) + h(e_2)$. Now, h realizes the ϕ_{ij} values on T^ρ. Thus, h^* also realizes the ϕ_{ij} values, on T and since (as we show) the edge weighting is strictly positive, it follows, by classical results [10], that this is the unique such edge weighting of T. Thus $\lambda = h^*$.

Now for an edge $e=(v,w)$ of T^ρ where w is a leaf,

$$h(e) \geqslant -\log\det(M(e)) \geqslant -0.5\log\det(M(e))$$

as claimed. Alternatively, for an edge $e=(v,w)$ of T for which neither of v, w are leaves, we have

$$h(e) = -\log\det(M(e)) - 0.5\log\left[\prod_\mu \pi(v)_\mu\right] + 0.5\log\left[\prod_\mu \pi(w)_\mu\right].$$

In order to derive our desired inequality we establish a further result. Let us suppose $M=[M_{\mu\nu}]$ is any $r\times r$ matrix with non-negative entries and $\boldsymbol{x}$ is a row vector of length r with non-negative entries. We claim that

$$\prod_\mu (\boldsymbol{x}M)_\mu \geqslant |\det(M)|\prod_\mu x_\mu.$$

To obtain this, note that the left-hand side is just

$$\prod_\mu \left(\sum_\nu x_\nu M_{\nu\mu}\right) \geqslant \left(\sum_\sigma M_{\sigma(1)1}M_{\sigma(2)2}\ldots M_{\sigma(r)r}\right)\prod_\mu x_\mu,$$

where the second summation is over all permutations σ of $(1,2,\ldots,r)$, and so this sum is at least $|\det(M)|$, since the permanent of a nonnegative matrix is never smaller than the absolute value of its determinant. Now, $[\pi(w)_1,\ldots,\pi(w)_r]=[\pi(v)_1,\ldots,\pi(v)_r]M(e)$, and so, applying the above inequality to the case $M=M(e)$ and $x=[\pi(v)_1,\ldots,\pi(v)_r]$, we obtain

$$\prod_\mu \pi(w)_\mu \geqslant \det(M(e))\prod_\mu \pi(v)_\mu.$$

Thus,

$$0.5\log\det(M(e)) \leqslant 0.5\log\left[\prod_\nu \pi(w)_\nu\right] - 0.5\log\left[\prod_\mu \pi(v)_\mu\right]$$

and so

$$\begin{aligned} h(e) &= -0.5\log\det(M(e)) \\ &\quad - 0.5\left(\log\det(M(e)) + \log\left[\prod_\mu \pi(v)_\mu\right] - \log\left[\prod_\mu \pi(w)_\mu\right]\right) \\ &\geqslant -0.5\log\det(M(e)), \end{aligned}$$

as claimed.

The inequalities for h now extend to $h^*=\lambda$ for all (non-troublesome) edges of T. If T^ρ has a troublesome edge e then $\lambda(e)=h^*(e)=h(e_1)+h(e_2)$, and from the above we have $h(e_i)\geqslant -0.5\log\det(M(e_i))$ for $i=1,2$. □

Theorem 14. *Let $x=x(n)$ and $x'=x'(n)$ be such that for all edges in the tree T, $0<1-2x'\leqslant \det(M(e))\leqslant 1-2x<1$. Assume x' has an upper bound strictly less than $1/2$. Mutatis mutandis, algorithms FPM, DCM and WAM, Theorems 9, 11, and 12 generalize to the general stochastic model under (40) and (41). WAM and DCM returns the binary model tree T with probability $1-o(1)$ if*

$$k>\frac{c\cdot\log n}{x^2(1-2x')^{4depth(T)}} \tag{43}$$

with a certain constant c.

Proof. Recall the definition of the corrected empirical distance, $\hat{\phi}_{ij}$, and $\delta(x)$ $(=-0.5\log(1-2x))$. We first establish the following

Claim: *If*

$$|\phi_{ij}-\hat{\phi}_{ij}|>\delta(x)/2 \tag{44}$$

then

$$|\det(F^{ij})-\det(\hat{F}^{ij})|>x\det(F^{ij})/4. \tag{45}$$

Proof of Claim: By inequality (44),

$$|\log(\det(F^{ij}))-\log(\det(\hat{F}^{ij}))|=\left|\log\left(\frac{\det(\hat{F}^{ij})}{\det(F^{ij})}\right)\right|>-\frac{1}{4}\log(1-2x)$$

and so $\det(\hat{F}^{ij})/\det(F^{ij})$ is either greater than $(1-2x)^{-1/4}$, or less than $(1-2x)^{1/4}$. Thus, $|\det(F^{ij})-\det(\hat{F}^{ij})|>\min\{\alpha^-(x),\alpha^+(x)\}\det(F^{ij})$ where $\alpha^+(x):=1-(1-2x)^{1/4}$; $\alpha^-=(1-2x)^{-1/4}-1$. Now, it can be checked that, for x strictly between 0 and $1/2$, $\alpha^-(x),\alpha^+(x)>x/4$ which establishes the Claim.

To apply Lemma 3, we need to know how $\det(\hat{F}^{ij})$ responds to the replacement at one site of a pattern by a different pattern. If $\hat{F}_1^{ij}$ is the resulting F-matrix for this perturbed data set, then

$$\hat{F}_1^{ij}=\hat{F}^{ij}+(1/k)D^{ij}$$

where D^{ij} has one entry of $+1$, one entry of -1, and all other entries 0. Consequently,

$$|\det(\hat{F}_1^{ij})-\det(\hat{F}^{ij})|\leqslant c_1/k \tag{46}$$

for some constant c_1.

Next, for any real analytic function f defined on a vector x having a normalized multinomial distribution with parameters k and μ, we have (by Taylor expansion of f about μ to the second derivative term, followed by application of the expectation operator):

$$|\mathbb{E}[f(x)]-f(\mu)|\leqslant\frac{1}{2}M\sum_{i,j}|cov(x_i,x_j)|,$$

where $cov(x_i,x_j)$ is the covariance of x_i,x_j (equal to $\mu_i(1-\mu_i)/k$, when $i=j$, and $-\mu_i\mu_j/k$ otherwise); and where M is the maximal value of any of the second derivatives of f over the unit simplex. Thus, since $\det(F^{ij})$ is a polynomial in the entries of F^{ij}, we have:

$$|\mathbb{E}[\det(\hat{F}^{ij})] - \det(F^{ij})| \leqslant c_2/k \tag{47}$$

for some constant c_2. Combining (47) with the triangle inequality gives

$$|\det(F^{ij}) - \det(\hat{F}^{ij})| \leqslant |\det(\hat{F}^{ij}) - \mathbb{E}[\det(\hat{F}^{ij})]| + c_2/k$$

and so

$$\mathbb{P}[|\det(F^{ij}) - \det(\hat{F}^{ij})| > t] \leqslant \mathbb{P}[|\det(\hat{F}^{ij}) - \mathbb{E}[\det(\hat{F}^{ij})]| > (t - c_2/k)] \tag{48}$$

for any $t>0$. Hence by Lemma 3, applied with (46), we have

$$\mathbb{P}[|\det(F^{ij}) - \det(\hat{F}^{ij})| > x\det(F^{ij})/4] \leqslant 2\exp\left(-d\left(\frac{x\det(F^{ij})}{4} - \frac{c_2}{k}\right)^2 k\right) \tag{49}$$

for a constant d. For the validity of the latter argument we need that

$$\frac{x\det(F^{ij})}{4} - \frac{c_2}{k} > 0. \tag{50}$$

Now, how can we set a lower bound for $\det(F^{ij})$? Note that $\det(F^{ij})$ is just the product of $\det(M(e))$ over all edges on the path from i to j, times the product of $\pi(v_{ij})_\mu$ over all states μ, where $\pi(v)$ is the vector of probabilities of states at vertex v, and v_{ij} is the most recent common ancestor of i and j in the tree. Due to our hypotheses (41), we have

$$\det F^{ij} > c_3(1-2x')^{d(i,j)} \tag{51}$$

with a positive constant c_3. However, the conditions of the Theorem required $k > cx^{-1}(1-2x')^{-d(i,j)}$, and therefore taking a sufficiently large c guarantees (50).

Putting the pieces (44), (45) and (49) together we see that

$$\mathbb{P}[|\phi_{ij} - \hat{\phi}_{ij}| > \delta(x)/2] \leqslant 2\exp\left(-d\left(\frac{x\det(F^{ij})}{4} - \frac{c_2}{k}\right)^2 k\right). \tag{52}$$

Combining (51) and (42), we have

$$\mathbb{P}\left[|\phi_{ij} - \hat{\phi}_{ij}| > (1/2)\min_e\{\lambda(e)\}\right] \leqslant 2\exp\left(-d\left(c_4\frac{x(1-2x')^{d(i,j)}}{4} - \frac{c_2}{k}\right)^2 k\right),$$

where c_4 is a positive constant, and $d(i,j)$ is the number of edges in T separating leaves i and j. Hence, for any fixed quartet q of diameter $diam(q)$,

$$\mathbb{P}[\text{FPM errs on } q] \leqslant K\exp(-D'x^2(1-2x')^{2diam(q)}k) \tag{53}$$

for constants D', K. Thus we have an analogue of Theorem 9.

Now we show how to generalize the proof of Theorem 11. To avoid needless repetitions, we give details for the proof of assumption (9) only, and leave the proofs of assumptions (10) and (11) to the Reader. Note that the proof of correctness of DCM hinges exactly on assumption (9). Having a distance function in the general model, the width and algorithmic operations based on width generalize in a straightforward way.

For k evolving sites (i.e. sequences of length k), and $\tau > 0$, let us define the following two sets, $S_\tau = \{\{i,j\}: \det(\hat{F}^{ij}) > 2\tau\}$ and $Z_\tau = \{q \in \binom{[n]}{4}: \text{for all } i,j \in q, \{i,j\} \in S_{2\tau}\}$ (note the similarity between the definition for the set Z_τ, and that for the set Q_w of quartet splits of quartets of width at most w). We also define the following two events, $A = \{Q_{\text{short}}(T) \subseteq Z_\tau\}$ and $B =$ FPM correctly reconstructs the tree for all $q \in Z_\tau$. Thus, $\mathbb{P}[\mathscr{A} \cap \mathscr{B} \neq \emptyset] \geqslant \mathbb{P}[A \cap B]$. Let C be the event: "$S_{2\tau}$ contains all pairs $\{i,j\}$ with $\det(F^{ij}) > 6\tau$, and no pair $\{i,j\}$ with $\det(F^{ij}) \leqslant 2\tau$". Define $\lambda = \varepsilon^r(1-2x')^{2depth(T)+3}$. We claim that

$$\mathbb{P}[C] \geqslant 1 - (n^2 - n)\mathrm{e}^{-c\tau^2 k} \tag{54}$$

for a constant $c > 0$ and

$$\mathbb{P}[A|C] = 1 \quad \text{if } \tau \leqslant \lambda/6. \tag{55}$$

Suppose $\det(F^{ij}) \leqslant 2\tau$. To establish (54), using arguments similar to those between (45) and (49) one easily sees that Lemma 3 applies and

$$\begin{aligned}\mathbb{P}[\{i,j\} \in S_{2\tau}] &= \mathbb{P}[\det(\hat{F}^{ij}) > 4\tau] \\ &\leqslant \mathbb{P}[\det(\hat{F}^{ij}) - \det(F^{ij}) \geqslant 2\tau] \leqslant \mathrm{e}^{-c\tau^2 k}\end{aligned}$$

for a constant $c > 0$.

Since there are at most $\binom{n}{2}$ such pairs $\{i,j\}$ such that $\det(F^{ij}) \leqslant 2\tau$, the probability that at least one such pair lies in $S_{2\tau}$ is at most $\binom{n}{2}\mathrm{e}^{-c\tau^2 k}$. By a similar argument, the probability that $S_{2\tau}$ fails to contain a pair $\{i,j\}$ with $\det(F^{ij}) > 6\tau$ is also at most $\binom{n}{2}\mathrm{e}^{-c\tau^2 k}$. These two bounds establish (54).

We now establish (55). For $q \in Q_{\text{short}}(T)$ and $i,j \in q$, if a path $e_1 e_2 \ldots e_t$ joins leaves i and j, then $t \leqslant 2depth(T)+3$ by the definition of $Q_{\text{short}}(T)$. Using these facts, and the bound $\det(M(e)) \geqslant 1-2x'$, we obtain $\det(F^{ij}) \geqslant \varepsilon^r(1-2x')^t$. Consequently, $\det(F^{ij}) > 6\tau$ (by assumption that $\tau \leqslant \lambda/6$) and so $\{i,j\} \in S_{2\tau}$ once we condition on the occurrence of event C. This holds for all $i,j \in q$, so by definition of Z_τ we have $q \in Z_\tau$. This establishes (55).

Then for any quartet $q \in Q_{\text{short}}(T)$, if e is the central edge of the contracted subtree induced by q in T, then $\det(M(e)) \leqslant 1 - 2x$. Furthermore, conditional on C, for any pendant edge $e, \det(M(e)) > \min\{\det(F^{ij}): i,j \in q\} > 2\tau$. Thus, by (53), which is the analogue of Theorem 9, and the Bonferroni inequality, we can follow the corresponding

proof from Theorem 11, to obtain (using (54) and (55))

$$\mathbb{P}[A \cap B] \geqslant 1 - K \binom{n}{4} \exp(-D'x^2(1-2x')^{2depth(T)+3}k) - (n^2 - n)e^{-d\lambda^2 k}$$

for constants $K, D' > 0$ Formula (43) now follows by an easy calculation.

Note that the proof also handled the problem that arises if some logarithms are to be taken of negative numbers and so we cannot even compute corrected distances. The morale is that those pairs are not needed according to the proof. Therefore there is no need for additional conditioning for the shape of the observed data.

8. Considerations for biological data analysis

The focus of this paper has been to establish analytically that every evolutionary tree is accurately reconstructable from quartets of closely related taxa, and, furthermore, this requires just very short sequences, given certain assumptions about the model tree. This is a significant theoretical result, especially since the bounds that we obtain indicate that the sequence lengths that suffices for accuracy with high probability using our new methods are very much shorter than those that suffice for accuracy using other very promising distance-based methods. However, are these observations significant for biological datasets? And if they are, are these methods likely to be practically useful (or merely indications of what might be achieved in future)?

The answer to the first question, concerning the significant for biological datasets, depends upon whether there are biologically realistic evolutionary trees that have smaller "weighted depth" than "weighted diameter", a concept that we now define.

Let T be an edge-weighted tree with positive weights on the internal edges and non-negative weights on the edges incident with leaves. Let e be an internal edge of the tree. The *weighted depth* around edge e is the minimum value of q so that there exists a set of four leaves, i, j, k, l, with one leaf in each of the four subtrees induced by the removal of e and its endpoints, where $q = \max\{d_{ij}^T, d_{ik}^T, d_{il}^T, d_{jk}^T, d_{jl}^T, d_{kl}^T\}$. The *weighted depth* of the tree T is then the maximum weighted depth of any edge in T. The *weighted diameter* of a tree T is simply the maximum d_{xy}^T, taken over all pairs of leaves x, y. We will denote the weighted depth of a tree T by $w\,depth(T)$ and its weighted diameter by $w\,diam(T)$.

The analysis given in the previous sections of the sequence length that suffices for accuracy for various methods can be restated as follows:

Corollary 1. *DCM and WAM will be accurate with probability* $1-\delta$ *if the sequence length exceeds*

$$c \log n e^{O(w\,depth(T))},$$

where c is a constant that depends upon only $f = \min_e p(e)$ *and* δ. *The other distance based methods (Agarwala et al.'s single-pivot algorithm and its variant, the double-*

pivot algorithm, the naive method, and neighbor-joining) are accurate with the same probability if the sequence length exceeds

$$c' \log n e^{O(w\,diam(T))},$$

where c' is a constant that also depends only upon $f = \min_e\{p(e)\}$ and δ.

These are only upper bounds (i.e. these may be loose, and exact accuracy may be possible from shorter sequences), but these are also currently the best upper bounds that are known for these methods, to our knowledge.

Thus, to compare the sequence lengths that suffice for exact topological accuracy, we need to compare the weighted depth to the weighted diameter. A reasonable comparison between these two quantities for biologically realistic trees is difficult, as there are very few well established evolutionary trees, especially of large divergent datasets. On the other hand, for some data sets, evolution may proceed in a more-or-less clock-like fashion (i.e. the number of mutations that occurs along an evolutionary lineage is roughly proportional to time). For such data sets, it can be seen that the weighted depth and the weighted diameter are exactly the same. Under these circumstances, there is no benefit to using DCM or WAM instead of one of the better other distance methods, such as neighbor joining, although this analysis also does not suggest that neighbor joining will outperform DCM or WAM (to be precise, the conditions that guarantee accuracy for neighbor-joining will also guarantee accuracy for DCM and WAM, and vice versa). Thus, for clock-like evolutionary conditions, these techniques do not provide any advantage from a theoretical standpoint.

On the other hand, there *are* important biological data sets for which evolution proceeds in a very non-clock like fashion, according to various analyses by biologists and statisticians (see, for example, [55, 56]). For these data sets, there *could* be significant advantage obtained by using techniques such as DCM and WAM, which examine only closely related taxa in order to reconstruct the tree. The degree to which DCM and WAM could provide an advantage would theoretically depend upon the magnitude of the difference between the weighted depth and weighted diameter. This magnitude is likely to be largest for sets of highly divergent taxa, rather than for closely related taxa.

As a practical tool, DCM and WAM are not entirely satisfactory, in part because DCM and WAM only return trees when the conditions hold for exact accuracy. Although some biologists would rather get no tree than get an incorrect tree [41], not all biologists share this view, and so partially correct trees are often desirable. Thus, the answer to the second question is basically negative.

However, DCM and WAM were not designed to be practical tools, but rather to indicate theoretical possibilities, and to suggest how better methods might be invented which could have the theoretical guarantees that DCM and WAM provide, while having better performance in practice. Furthermore, such methods *have* recently been developed. The *disk-covering method* of Huson et al. [36] the *harmonic greedy triples method* of Csuros and Kao [16], and the method of Cryan et al. [15] have each used

the observations in this paper and obtained methods with convergence rates that are never worse than polynomial by using only small distances to (re)construct the tree.

Acknowledgements

We thank Scott Nettles for fruitful discussions of efficient implementations, and David Bryant and Éva Czabarka for proof reading the manuscript. We also thank A. Ambainis and another (anonymous) referee, for their careful reading of the manuscript and helpful suggestions for improving the exposition.

Péter L. Erdős was supported in part by the Hungarian National Science Fund contract T 016 358. László A. Székely was supported by the National Science Foundation grant DMS 9701211, the Hungarian National Science Fund contracts T 016 358 and T 019 367, and European Communities (Cooperation in Science and Technology with Central and Eastern European Countries) contract ERBCIPACT 930 113. Michael A. Steel was supported by the New Zealand Marsden Fund and the New Zealand Ministry of Research, Science and Technology. Tandy J. Warnow was supported by an NSF Young Investigator Award CCR-9457800, a David and Lucille Packard Foundation fellowship, and generous research support from the Penn Research Foundation and Paul Angello.

This research started in 1995 when the authors enjoyed the hospitality of DIMACS during the Special Year for Mathematical Support to Molecular Biology, and was completed in 1997 while enjoying the hospitality of Andreas Dress, at Universität Bielefeld, in Germany.

References

[1] R. Agarwala, V. Bafna, M. Farach, B. Narayanan, M. Paterson, M. Thorup, On the approximability of numerical taxonomy: fitting distances by tree metrics, in: Proc. 7th Annual ACM–SIAM Symp. on Discrete Algorithms, 1996, pp. 365–372.

[2] D.J. Aldous, Probability distributions on cladograms, in: D.J. Aldous, R. Permantle (Eds.), Discrete Random Structures, IMA Volume in Mathematics and its Applications, vol. 76, Springer, Berlin, 1995, pp. 1–18.

[3] N. Alon, J.H. Spencer, The Probabilistic Method, Wiley, New York, 1992.

[4] A. Ambainis, R. Desper, M. Farach, S. Kannan, Nearly tight bounds on the learnability of evolution, in: 38th Annual Symp. on Foundations of Computer Science, Miami Beach, FL, 20–22 October 1997. IEEE Science, New York, pp. 524–533.

[5] K. Atteson, The performance of neighbor-joining algorithms of phylogeny reconstruction, in: Computing and Combinatorics, 3rd Annual Internat. Conf., COCOON'97, Shanghai, China, August 1997, COCOON'97, Lecture Notes in Computer Science, vol. 1276, Springer, Berlin, pp. 101–110.

[6] H.-J. Bandelt, A. Dress, Reconstructing the shape of a tree from observed dissimilarity data, Adv. Appl. Math. 7 (1986) 309–343.

[7] V. Berry, O. Gascuel, Inferring evolutionary trees with strong combinatorial evidence, in: Computing and Combinatorics, 3rd Annual Internat. Conf., COCOON'97, Shanghai, China, August 1997, COCOON'97, Lecture Notes in Computer Science, vol. 1276, Springer, Berlin, pp. 111–123.

[8] J.K.M. Brown, Probabilities of evolutionary trees, Syst. Biol. 43 (1994) 78–91.

[9] D.J. Bryant, M.A. Steel, Extension operations on sets of leaf-labelled trees, Adv. Appl. Math. 16 (1995) 425–453.

[10] P. Buneman, The recovery of trees from measures of dissimilarity, in: F.R. Hodson, D.G. Kendall, P. Tautu (Eds.), Mathematics in the Archaeological and Historical Sciences, Edinburgh University Press, Edinburgh, 1971, pp. 387–395.
[11] M. Carter, M. Hendy, D. Penny, L.A. Székely, N.C. Wormald, On the distribution of lengths of evolutionary trees, SIAM J. Discrete Math. 3 (1990) 38–47.
[12] J.A. Cavender, Taxonomy with confidence, Math. Biosci. 40 (1978) 271–280.
[13] J.T. Chang, J.A. Hartigan, Reconstruction of evolutionary trees from pairwise distributions on current species, in: Computing Science and Statistics: Proc. 23rd Symp. on the Interface, 1991, pp. 254–257.
[14] H. Colonius, H.H. Schultze, Tree structure for proximity data, Br. J. Math. Statist. Psychol. 34 (1981) 167–180.
[15] M. Cryan, L.A. Goldberg, P.W. Goldberg, Evolutionary trees can be learned in polynomial time in the two-state general Markov-model, in: Proc. 39th Annual IEEE Symp. on Foundations of Computer Science, 1998, pp. 436–445.
[16] M. Csuros, M.-Y. Kao, Fast reconstruction of evolutionary trees through Harmonic Greedy Triplets, in: Proc. ACM–SIAM Symp. on Discrete Algorithms, 1999, pp. 261–268.
[17] W.H.E. Day, Computational complexity of inferring phylogenies from dissimilarities matrices, Inform. Process. Lett. 30 (1989) 215–220.
[18] P.L. Erdős, M.A. Steel, L.A. Székely, T. Warnow, Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule, Comput. Artif. Intell. 16 (2) (1997) 217–227.
[19] P.L. Erdős, M.A. Steel, L.A. Székely, T. Warnow, Inferring big trees from short quartets, in: ICALP'97, 24th Internat. Colloquium on Automata, Languages, and Programming, Silver Jubilee of EATCS, Bologna, Italy, July 7–11, 1997, Lecture Notes in Computer Science, vol. 1256, Springer, Berlin, 1997, pp. 1–11.
[20] P.L. Erdős, M.A. Steel, L.A. Székely, T. Warnow, A few logs suffice to build (almost) all trees (I), Random Struct. Algorithms 14 (1999) 153–184.
[21] M. Farach, J. Cohen, Numerical taxonomy on data: experimental results, in: Proc. ACM–SIAM Symp. on Discrete Algorithms, 1997.
[22] M. Farach, S. Kannan, Efficient algorithms for inverting evolution, in: Proc. ACM Symp. on the Foundations of Computer Science, 1996, pp. 230–236.
[23] M. Farach, S. Kannan, T. Warnow, A robust model for inferring optimal evolutionary trees, Algorithmica 13 (1995) 155–179.
[24] J.S. Farris, A probability model for inferring evolutionary trees, Syst. Zool. 22 (1973) 250–256.
[25] J. Felsenstein, Cases in which parsimony or compatibility methods will be positively misleading, Syst. Zool. 27 (1978) 401–410.
[26] J. Felsenstein, Numerical methods for inferring evolutionary trees, Quart. Rev. Biol. 57 (1982) 379–404.
[27] D. Feng, R. Doolittle, Progressive sequence alignment as a prerequisite to correct phylogenetic trees, J. Mol. Evol. 25 (1987) 351–360.
[28] Green Plant Phylogeny Research Coordination Group, Summary Report of Workshop #1: Current Status of the Phylogeny of the Charophyte Green Algae and the Embryophytes (University and Jepson Herbaria, University of California, Berkeley, June 24–28, 1995), 1996.
[29] E.F. Harding, The probabilities of rooted tree shapes generated by random bifurcation, Adv. Appl. Probab. 3 (1971) 44–77.
[30] J. Hein, A new method that simultaneously aligns and reconstructs ancestral sequences for any number of homologous sequences, when the phylogeny is given, Mol. Biol. Evol. 6 (1989) 649–668.
[31] D. Hillis, Approaches for assessing phylogenetic accuracy, Syst. Biol. 44 (1995) 3–16.
[32] D. Hillis, Inferring complex phylogenies, Nature 383 (1996) 130–131.
[33] P. Hogeweg, B. Hesper, The alignment of sets of sequences and the construction of phylogenetic trees, an integrated method, J. Mol. Evol. 20 (1984) 175–186.
[34] J. Huelsenbeck, Performance of phylogenetic methods in simulation, Syst. Biol. 44 (1995) 17–48.
[35] J.P. Huelsenbeck, D. Hillis, Success of phylogenetic methods in the four-taxon case, Syst. Biol. 42 (1993) 247–264.
[36] D. Huson, S. Nettles, T. Warnow, Inferring very large evolutionary trees from very short sequences, Proc., RECOMB, 1999.
[37] T. Jiang, E. Lawler, L. Wang, Aligning sequences via an evolutionary tree: complexity and approximation ACM STOC'94.
[38] S. Kannan, personal communication.

[39] M. Kimura, Estimation of evolutionary distances between homologous nucleotide sequences, Proc. Natl. Acad. Sci. USA 78 (1981) 454–458.
[40] J. Neyman, Molecular studies of evolution: a source of novel statistical problems, in: S.S. Gupta, J. Yackel (Eds.), Statistical Decision Theory and Related Topics, Academic Press, New York, 1971, pp. 1–27.
[41] K. Rice, personal communication.
[42] K. Rice, T. Warnow, Parsimony is hard to beat, in: T. Jiang, D.T. Lee (Eds.), COCOON'97, Computing and Combinatorics, 3rd Annual Internat. Conf., Shanghai, August 20–22, 1997, Lecture Notes in Computer Science, vol. 1276, Springer, Berlin, pp. 124–133.
[43] N. Saitou, M. Nei, The neighbor-joining method: a new method for reconstructing phylogenetic trees, Mol. Biol. Evol. 4 (1987) 406–425.
[44] N. Saitou, T. Imanishi, Relative efficiencies of the Fitch-Margoliash, maximum parsimony, maximum likelihood, minimum evolution, and neighbor-joining methods of phylogenetic tree construction in obtaining the correct tree, Mol. Biol. Evol. 6 (1987) 514–525.
[45] Y.A. Smolensky, A method for linear recording of graphs, USSR Comput. Math. Phys. 2 (1969) 396–397.
[46] M.A. Steel, The complexity of reconstructing trees from qualitative characters and subtrees, J. Classification 9 (1992) 91–116.
[47] M.A. Steel, Recovering a tree from the leaf colourations it generates under a Markov model, Appl. Math. Lett. 7 (1994) 19–24.
[48] M. Steel, M.D. Hendy, D. Penny, Reconstructing phylogenies from nucleotide pattern probabilities: a survey and some new results, Discrete Appl. Math. 89 (1999) 367–396.
[49] M.A. Steel, L.A. Székely, M.D. Hendy, Reconstructing trees when sequence sites evolve at variable rates, J. Comput. Biol. 1 (1994) 153–163.
[50] K. Strimmer, A. von Haeseler, Quartet Puzzling: a quartet Maximum Likelihood method for reconstructing tree topologies, Mol. Biol. Evol. 13 (1996) 964–969.
[51] K. Strimmer, N. Goldman, A. von Haeseler, Bayesian probabilities and Quartet Puzzling, Mol. Biol. Evol. 14 (1997) 210–211.
[52] D.L. Swofford, G.J. Olsen, P.J. Waddell, D.M. Hillis, Phylogenetic inference, Ch. 11, in: D.M. Hillis, C. Moritz, B.K. Mable (Eds.), Molecular Systematics, 2nd ed., Sinauer Associates, Inc., Sunderland, 1996, pp. 407–514.
[53] D. Harel, R.E. Tarjan, Fast algorithms for finding nearest common ancestors, SIAM J. Comput. 13 (1984) 338–355.
[54] N. Takezaki, M. Nei, Inconsistency of the maximum parsimony method when the rate of nucleotide substitution is constant, J. Mol. Evol. 39 (1994) 210–218.
[55] L. Vawter, W. Brown, Nuclear and mitochondrial DNA comparisons reveal extreme rate variation in the molecular clock, Science 234 (1986) 194–196.
[56] L. Vawter, W. Brown, Rates and patterns of base change in the small subunit ribosomal RNA gene, Genetics 134 (1993) 597–608.
[57] L. Wang, D. Gusfield, Improved approximation algorithms for tree alignment, J. Algorithms 25 (1997) 255–273.
[58] L. Wang, T. Jiang, On the complexity of multiple sequence alignment, J. Comput. Biol. 1 (1994) 337–348.
[59] L. Wang, T. Jiang, E. Lawler, Approximation algorithms for tree alignment with a given phylogeny, Algorithmica 16 (1996) 302–315.
[60] L. Wang, T. Jiang, D. Gusfield, A more efficient approximation scheme for tree alignment, in: Proc. 1st Annual Internat. Conf. on Computational Molecular Biology, January 1997, Santa Fe, NM, USA.
[61] T. Warnow, Combinatorial algorithms for constructing phylogenetic trees, Ph.D. Thesis, University of California-Berkeley, 1991.
[62] M.S. Waterman, T.F. Smith, M. Singh, W.A. Beyer, Additive evolutionary trees, J. Theoret. Biol. 64 (1977) 199–213.
[63] A.C. Wilson, R.L. Cann, The recent African genesis of humans, Scientific Amer. 266 (1992) 68–73.
[64] K.A. Zaretsky, Reconstruction of a tree from the distances between its pendant vertices, Uspekhi Math. Nauk (Russian Mathematical Surveys) 20 (1965) 90–92 (in Russian).

Annals of Combinatorics 7 (2003) 155-169
0218-0006/03/020155-15
DOI 10.1007/s00026-003-0179-x

Annals of Combinatorics

X-Trees and Weighted Quartet Systems

Andreas W.M. Dress[1*] and Péter L. Erdős[2†]

[1]Forschungsschwerpunkt Mathematisierung-Struktubildungprozesse, University of Bielefeld
P.O. Box 100131, 33501 Bielefeld, Germany
dress@mathematik.uni-bielefeld.de

[2]A. Rényi Institute of Mathematics, Hungarian Academy of Sciences, Budapest, P.O. Box 127
1364 Hungary
elp@renyi.hu

Received April 17, 2003

AMS Subject Classification: 05C05, 92D15, 92B05

Abstract. In this note, we consider a finite set X and maps W from the set $\mathcal{S}_{2|2}(X)$ of all 2, 2-splits of X into $\mathbb{R}_{\geq 0}$. We show that such a map W is induced, in a canonical way, by a binary X-tree for which a positive length $\ell(e)$ is associated to every inner edge e if and only if (i) exactly two of the three numbers $W(ab|cd), W(ac|bd)$, and $W(ad|cb)$ vanish, for any four distinct elements a, b, c, d in X, (ii) $a \neq d$ and $W(ab|xc) + W(ax|cd) = W(ab|cd)$ holds for all a, b, c, d, x in X with $\#\{a, b, c, x\} = \#\{b, c, d, x\} = 4$ and $W(ab|cx), W(ax|cd) > 0$, and (iii) $W(ab|uv) \geq \min\big(W(ab|uw), W(ab|vw)\big)$ holds for any five distinct elements a, b, u, v, w in X. Possible generalizations regarding arbitrary $\mathbb{R}$-trees and applications regarding tree-reconstruction algorithms are indicated.

Keywords: biological systematics, phylogeny, phylogenetic combinatorics, evolutionary trees, tree reconstruction, X-trees, quartet methods, quartet systems, weighted quartet systems.

1. Introduction

Let X be a finite set of cardinality n, and let $T = (V, E)$ be an X-tree, i.e., a finite tree without vertices of degree 2 whose set of leaves coincides with X. Further,

(i) let $\binom{X}{i}$ denote, for any natural number i, the set of all subsets of X of cardinality i,
(ii) let $\mathcal{S}_{2|2}(X)$ denote the set of all *partial* 2, 2-*splits* of X:

$$\mathcal{S}_{2|2}(X) := \left\{ \left\{ \{a, b\}, \{c, d\} \right\} \middle| \{a, b\}, \{c, d\} \in \binom{X}{2};\ \{a, b\} \cap \{c, d\} = \emptyset \right\},$$

* Supported in part by the DFG.

† Supported by the Alexander v. Humboldt Stiftung and by the Hungarian NSF, under contract Nos. T34702, T37846.

(iii) let $E_0 = E_0(T)$ denote the set of *pending* edges of T, i.e., of edges incident with a leaf:

$$E_0 = E_0(T) := \{e \in E \,\big|\, e \cap X \neq \emptyset\},$$

(iv) let $E_1 = E_1(T)$ denote the complementary set of *inner* edges of T:

$$E_1 = E_1(T) := E \setminus E_0,$$

(v) and let

$$\ell\colon E_1 \to \mathbb{R}_{>0}$$

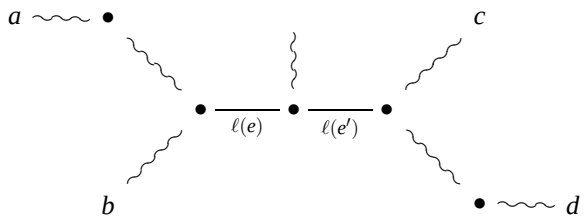
denote an arbitrary, but strictly positive *length function* defined on that set.

For convenience, we will also write $ab|cd$ for the unordered pair $\{\{a,b\},\{c,d\}\}$ of subsets of X of cardinality at most 2, for any $a,b,c,d \in X$ (so that $ab|cd \in \mathcal{S}_{2|2}(X)$ holds if and only if one has $\#\{a,b,c,d\} = 4$).

We are interested in the map $W = W_{T,\ell}$ defined on $\mathcal{S}_{2|2}(X)$ by

$$W\colon \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0},\ ab|cd \mapsto \sum_{e \in E(ab|cd)} \ell(e), \tag{1.1}$$

where the sum runs over the set $E(ab|cd)$ of all edges $e \in E$ that separate the leaves a, b from the leaves c, d. Clearly, the function W measures the total length of the "inner path" of the *quartet tree* $T_{a,b,c,d}$ "spanned" by a, b, c, d in case T contains at least one edge that separates a, b from c, d, and it vanishes otherwise.



The following facts are easily established:

(F1) Given any 4-subset $\{a,b,c,d\}$ of X, at least two of the three numbers $W(ab|cd)$, $W(ac|bd)$, and $W(ad|cb)$ vanish.

(F2) If T is binary, i.e., if all vertices in V outside X have degree 3 or — equivalently — if $\#V = 2n-2$ holds (recall that there is no vertex of degree 2), one has

$$W(ab|cd) + W(ac|bd) + W(ad|cb) > 0 \tag{1.2}$$

for all $\{a,b,c,d\} \in \binom{X}{4}$.

(F3) Given $a,b,c,d,x \in X$ with $\#\{a,b,c,x\} = \#\{b,c,d,x\} = 4$ and

$$W(ab|xc), W(bx|cd) > 0,$$

one has $\#\{a,b,c,d,x\} = 5$ and

$$W(ab|xc) + W(bx|cd) = W(ab|cd). \tag{1.3}$$

(F4) Given any 5-subset $\{a, b, u, v, w\}$ of X, one has

$$W(ab|uw) \geq \min\Big(W(ab|uv), W(ab|vw)\Big), \tag{1.4}$$

i.e., the two smaller ones of the three numbers

$$W(ab|uv), W(ab|uw), W(ab|vw)$$

must coincide or, still in other words, $W(ab|uv) < W(ab|uw)$ implies that $W(ab|uv) = W(ab|vw)$ for all $a, b, u, v, w \in X$ as above.

Our main result is the following:

Theorem 1.1. *A map*

$$W\colon \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$$

is of the form $W_{T,\ell}$ for some finite binary tree T with leave set X and some length function ℓ defined on the set $E_1(T)$ of inner edges of T if and only if W satisfies the conditions (F1) *to* (F4) *above. Moreover, if W satisfies those four conditions, the tree T and the length function $\ell\colon E_1(T) \to \mathbb{R}_{>0}$ with $W = W_{T,\ell}$ are uniquely determined* (*up to canonical isomorphism*) *by W.*

It was established already in 1977 by the psychologists Colonius and Schulze (cf. [5, 6]), the first two papers on quartet analysis that initiated much further work devoted to this topic, cf. [7–39] that, given any subset $\mathcal{Q}$ of $\mathcal{S}_{2|2}(X)$, there exists a binary X-tree $T = (V, E)$ such that the set

$$\mathcal{Q}_T := \big\{ab|cd \in \mathcal{S}_{2|2}(X) \;\big|\; E(ab|cd) \neq \emptyset\big\}$$

of 2|2-splits in $\mathcal{S}_{2|2}(X)$ induced by T coincides with $\mathcal{Q}$ if and only if the following three assertions hold:

(Q1) $\#(\mathcal{Q} \cap \{ab|cd, ac|bd, ad|cb\}) = 1$ holds for all $\{a, b, c, d\} \in \binom{X}{4}$,
(Q2) $ab|cx \in \mathcal{Q}$ and $ax|cd \in \mathcal{Q}$ implies $ab|cd \in \mathcal{Q}$ for all $\{a, b, c, d, x\} \in \binom{X}{5}$,
(Q3) $ab|uv, ab|vw \in \mathcal{Q}$ implies $ab|uw \in \mathcal{Q}$ for all $\{a, b, u, v, w\} \in \binom{X}{5}$,

in which case this tree is uniquely determined by $\mathcal{Q}$.

Thus, the *support*

$$\operatorname{supp}(W) := \big\{ab|cd \in \mathcal{S}_{2|2}(X) \;\big|\; W(ab|cd) \neq 0\big\}$$

of any map $W\colon \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$ that satisfies the conditions (F1) to (F4) above is obviously of the form $\mathcal{Q}_T$ for some unique binary X-tree T. Thus, a proof of the existence part of Theorem 1.1 could easily be based on this observation. In this note however, we want to proceed in a more direct way, not so much to avoid referring to any previous work, but because our direct approach also yields new tree-building strategies.

The paper is organized as follows: In the next section, we will show that the map $W_{T,\ell}\colon \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$ associated with a binary X-tree T and a length function $\ell\colon E_1(T) \to \mathbb{R}_{>0}$ determines T and ℓ up to canonical isomorphism. Then, we will show that a map $W\colon \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$ is of the form $W = W_{T,\ell}$ for some binary X-tree T and length function $\ell\colon E_1(T) \to \mathbb{R}_{>0}$ if and only if W satisfies the conditions (F1) to (F4) above. And finally, we shall discuss various promising directions of future research as well as some simple algorithmic applications of our results in the last section.

2. $W_{T,\ell}$ Determines T and ℓ up to Canonical Isomorphism

Given any two binary X-trees T and T' and maps $\ell\colon E_1(T) \to \mathbb{R}_{>0}$ and $\ell'\colon E_1(T') \to \mathbb{R}_{>0}$, we will show here that $W_{T,\ell} = W_{T',\ell'}$ implies the existence of a unique map $\varphi\colon V(T) \to V(T')$ with $\varphi(x) = x$ for all $x \in X$ and $\{\varphi(u), \varphi(v)\} \in E(T')$ for all $\{u, v\} \in E(T)$, and that this map is necessarily bijective, induces a bijection between $E(T)$ and $E(T')$, and commutes with ℓ and ℓ' (i.e., $\ell(\{u, v\}) = \ell'(\{\varphi(u), \varphi(v)\})$ holds for this map φ and all $\{u, v\} \in E(T)$).

To construct $\varphi(v)$, recall the following facts:

i) Given any finite connected graph $G = (V, E)$, the *standard* graph metric d_G induced on V by G is defined to be the map from $V \times V$ into $\mathbb{N}_0$ that maps each pair $(u, v) \in V \times V$ onto the minimal number $d_G(u, v)$ of edges that constitute a path from u to v in G, i.e., onto the minimum of all $k \in \mathbb{N}_0$ for which vertices $v_0 := u, v_1, \ldots, v_k := v \in V$ exist with $\{v_{i-1}, v_i\} \in E$ for all $i = 1, \ldots, k$.

ii) A finite connected graph $G = (V, E)$ is defined to be a *median* graph if, for all $u, v, w \in V$, there exists a unique vertex $m = \mathrm{med}_G(u, v, w)$ in V with

$$\begin{aligned} d_G(u, v) &= d_G(u, m) + d_G(m, v), \\ d_G(u, w) &= d_G(u, m) + d_G(m, w), \end{aligned}$$

and

$$d_G(v, w) = d_G(v, m) + d_G(m, w),$$

in which case $\mathrm{med}_G(u, v, w) = \mathrm{med}_G(v, u, w) = \mathrm{med}_G(u, w, v)$ and $\mathrm{med}_G(u, u, w) = u$ hold for all $u, v, w \in V$ (cf. [1]).

iii) Any X-tree $T = (V, E)$ is a median graph and every vertex v in V is of the form $v = \mathrm{med}_T(a, b, c)$ for some appropriate leaves a, b, c in X, and one has $\mathrm{med}_T(a, b, c) \in V - X$ for some $a, b, c \in X$ if and only if $\#\{a, b, c\} = 3$ holds.

iv) Given a X-tree $T = (V, E)$, a length function $\ell\colon E_1(T) \to \mathbb{R}_{>0}$, and four distinct leaves $a, b, c, d \in X$, one has $W_{T,\ell}(ab|cd) > 0$ if and only if one has

$$\mathrm{med}_T(a, b, c) = \mathrm{med}_T(a, b, d) \neq \mathrm{med}_T(a, c, d) = \mathrm{med}_T(b, c, d),$$

in which case $E(ab|cd)$ consists exactly of the set of edges $e \in E_1(T)$ on the unique path from $\mathrm{med}_T(a, b, c) = \mathrm{med}_T(a, b, d)$ to $\mathrm{med}_T(a, c, d) = \mathrm{med}_T(b, c, d)$ and $W_{T,\ell}(ab|cd)$ is exactly the length of that path relative to ℓ.

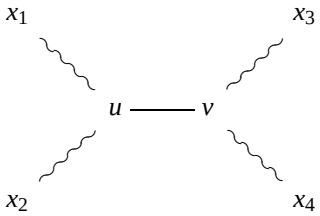
v) If, moreover, T is binary, one has

$$\mathrm{med}_T(a_1, a_2, a_3) = \mathrm{med}_T(b_1, a_2, a_3)$$

for four distinct elements $a_1, a_2, a_3, b_1 \in X$ if and only if one has $W_{T,\ell}(a_1b_1|a_2a_3) > 0$, and one has $\mathrm{med}_T(a_1, a_2, a_3) = \mathrm{med}_T(b_1, b_2, b_3)$ for some $a_1, a_2, a_3, b_1, b_2, b_3$ in X with $\#\{a_1, a_2, a_3\} = 3$ if and only if there exists a permutation π of the index set $\{1, 2, 3\}$ with either $a_i = b_{\pi(i)}$ or $\#\{a_1, a_2, a_3, b_{\pi(i)}\} = 4$ and $W_{T,\ell}(a_ib_{\pi(i)}|a_ja_k) > 0$ for all i, j, k in $\{1, 2, 3\}$ with $\{1, 2, 3\} = \{i, j, k\}$ in which case we must also have $\#\{b_1, b_2, b_3\} = 3$ as well as either $b_i = a_{\pi^{-1}(i)}$ or $\#\{b_1, b_2, b_3, a_{\pi^{-1}(i)}\} = 4$ and $W_{T,\ell}(b_ia_{\pi^{-1}(i)}|b_jb_k) > 0$ for all $i, j, k \in \{1, 2, 3\}$ with $\{1, 2, 3\} = \{i, j, k\}$.

In particular, we can decide whether we have $\mathrm{med}_T(a_1, a_2, a_3) = \mathrm{med}_T(b_1, b_2, b_3)$ for some $a_1, a_2, a_3, b_1, b_2, b_3$ in X with $\#\{a_1, a_2, a_3\} = 3$ from exclusively analysing the support of $W_{T,\ell}$.

vi) One can decide whether two distinct vertices u and v in T form an edge by studying medians: Indeed, given any two distinct vertices $u, v \in V$, one can choose elements $x_1, x_2, x_3, x_4 \in X$, not necessarily distinct, as indicated in the figure below:



i.e., with

$$u = \mathrm{med}_T(x_1, x_2, x_3) = \mathrm{med}_T(x_1, x_2, x_4)$$

and

$$v = \mathrm{med}_T(x_1, x_3, x_4) = \mathrm{med}_T(x_2, x_3, x_4),$$

and one has $\{u, v\} \in E(T)$ if and only if

$$\mathrm{med}_T(x_1, x_3, y) \in \big\{\mathrm{med}_T(x_1, x_2, y), \mathrm{med}_T(x_3, x_4, y), u, v\big\}$$

holds for all $y \in X$.

These well-known and easily established facts allow us to define the required map $\varphi\colon V(T) \to V(T')$: For every $x \in X$, we put $\varphi(x) := x$, and for every $v \in V(T) - X$, we choose $a_1, a_2, a_3 \in X$ with $v = \mathrm{med}_T(a_1, a_2, a_3)$ and put

$$\varphi(v) := \mathrm{med}_{T'}(a_1, a_2, a_3).$$

This is clearly well defined in view of Assertion v) above, we have $\varphi(x) = x$ for every $x \in X$ simply by definition, and we have

$$\varphi(v) = \mathrm{med}_{T'}(a_1, a_2, a_3)$$

for *all* $v \in V$ and $a_1, a_2, a_3 \in X$ with $v = \mathrm{med}_T(a_1, a_2, a_3)$ — even in case $v \in X$ because this implies that at least two of the three elements a_1, a_2, a_3 must coincide with v which in turn implies that

$$\mathrm{med}_{T'}(a_1, a_2, a_3) = v = \varphi(v)$$

must hold also in this case. Further, we have $\{\varphi(u), \varphi(v)\} \in E(T')$ for all $\{u, v\} \in E(T)$: Indeed, if $\{u, v\} \in E(T)$ holds, we can choose $x_1, x_2, x_3, x_4 \in X$ as described in Assertion vi) above and, applying φ, we get

$$\begin{aligned}\varphi(u) &= \mathrm{med}_{T'}(x_1, x_2, x_3) = \mathrm{med}_{T'}(x_1, x_2, x_4),\\ \varphi(v) &= \mathrm{med}_{T'}(x_1, x_3, x_4) = \mathrm{med}_{T'}(x_2, x_3, x_4),\end{aligned}$$

as well as

$$\begin{aligned}\mathrm{med}_{T'}(x_2,x_3,y) &= \varphi(\mathrm{med}_T(x_2,x_3,y))\\ &\in \big\{\varphi(\mathrm{med}_T(x_1,x_2,y)),\varphi(\mathrm{med}_T(x_2,x_3,y)),\varphi(u),\varphi(v)\big\}\\ &= \big\{\mathrm{med}_{T'}(x_1,x_2,y),\mathrm{med}_{T'}(x_2,x_3,y),\varphi(u),\varphi(v)\big\}\end{aligned}$$

for all $y \in X$. Hence,

$$\{\varphi(u),\varphi(v)\} \in E(T'),$$

as claimed.

It is also easy to see that any map $\varphi\colon V(T) \to V(T')$ with $\varphi(x) = x$ for all $x \in X$ and $\{\varphi(u),\varphi(v)\} \in E(T')$ for all $\{u,v\} \in E(T)$ is necessarily bijective and induces a bijection between $E(T)$ and $E(T')$ and, hence, also one between $E_1(T)$ and $E_1(T')$: Indeed, the image $\varphi(V(T))$ of $V(T)$ must contain all vertices on all paths between any two leaves in T', and the image $\big\{\{\varphi(u),\varphi(v)\}\,\big|\,\{u,v\} \in E(T)\big\}$ of $E(T)$ must contain all edges on all of those paths. Thus, the map $\varphi\colon V(T) \to V(T')$ as well as the induced map from $E(T)$ into $E(T')$ must be surjective and, hence, bijective because one has $\#V(T) = \#V(T') = 2n-2$ and $\#E(T) = \#E(T') = \#V(T) - 1 = 2n-3$ in view of the fact that both, T and T', were assumed to be binary X-trees.

Finally, we have necessarily

$$\ell(\{u,v\}) = \ell'(\{\varphi(u),\varphi(v)\})$$

for any edge $\{u,v\} \in E_1$ because, as above, we can choose $x_1,x_2,x_3,x_4 \in X$ with $u = \mathrm{med}_T(x_1,x_2,x_3) = \mathrm{med}_T(x_2,x_2,x_3)$ and $v = \mathrm{med}_T(x_2,x_3,x_4) = \mathrm{med}_T(x_1,x_3,x_4)$. Hence,

$$\ell(\{u,v\}) = W_{T,\ell}(x_1x_2\big|x_3x_4) = W_{T',\ell'}(x_1x_2\big|x_3x_4) = \ell'(\{\varphi(u),\varphi(v)\}),$$

as claimed.

It remains to observe that φ is uniquely determined by T and T': However, as observed already above, any map $\psi\colon V(T) \to V(T')$ with $\psi(x) = x$ for all $x \in X$ and $\{\psi(u),\psi(v)\} \in E(T')$ for all $\{u,v\} \in E(T)$ is necessarily bijective and induces a bijection from $E(T)$ onto $E(T')$. Thus, $d_T(x,y) = d_{T'}(x,y)$, and hence,

$$\psi(\mathrm{med}_T(x,y,z)) = \mathrm{med}_{T'}(x,y,z) = \varphi(\mathrm{med}_T(x,y,z))$$

must hold for all $x,y,z \in X$ implying that also $\psi(v) = \varphi(v)$ must hold for all $v \in V$.

3. Deriving T and ℓ from W

In this section, we will assume throughout that W is a map from $\mathcal{S}_{2|2}(X)$ into $\mathbb{R}_{\geq 0}$ that satisfies the conditions (F1) to (F4) stated above, and we want to show that a binary X-tree T and a map $\ell\colon E_1(T) \to \mathbb{R}_{>0}$ with $W = W_{T,\ell}$ then necessarily exist.

To simplify notations, we will say that $W(ab|x|cd)$ holds for some elements a, b, c, d, x in X if and only if the four elements a, b, x, c and the four elements b, x, c, d are distinct and one has $W(ab|xc), W(bx|cd) > 0$. We will begin by collecting some technicalities regarding this quinternary relation. Note first that $W(ab|x|cd)$ implies $\#\{a, b, x, c, d\} = 5$ and

$$W(ab|cd) = W(ab|xc) + W(bx|cd) > W(ab|xc), W(bx|cd) > 0$$

in view of (F3). Hence,

$$W(ab|xc) = W(ab|xd) > 0,\ W(ax|cd) = W(bx|cd) > 0 \tag{3.1}$$

in view of (F4). This proves the implication "**(i)** $\Rightarrow$ **(ii)**" in

Lemma 3.1. *For all a, b, c, d, x in X, the following assertions are equivalent*:

- **(i)** $W(ab|x|cd)$ *holds, i.e., one has* $\#\{a, b, x, c\} = \#\{b, x, c, d\} = 4$ *and* $W(ab|xc)$, $W(bx|cd) > 0$.
- **(ii)** $\#\{a, b, x, c, d\} = 5$, $W(ab|cd) = W(ab|xc) + W(bx|cd)$, $W(ab|xd) = W(ab|xc) > 0$, *and* $W(ax|cd) = W(bx|cd) > 0$.
- **(iii)** $\#\{a, b, c, d\} = \#\{a, b, d, x\} = 4$ *and* $W(ab|cd) > W(ab|xd) > 0$.
- **(iv)** $\#\{a, b, x, c, d\} = 5$, $W(ab|cd) > 0$, $W(ab|xc) = W(ab|xd)$, *furthermore* $W(xa|dc) = W(xb|dc)$.

In particular, given any 5*-subset* $\{a, b, x, c, d\}$ *of* X, *one has*

$$W(ab|x|cd) \Leftrightarrow W(ba|x|cd) \Leftrightarrow W(cd|x|ab) \Leftrightarrow \cdots.$$

Proof. It is obvious that **(ii)** $\Rightarrow$ **(iii)** and **(ii)** $\Rightarrow$ **(iv)** hold.

(iii) $\Rightarrow$ **(i)**: Clearly, we must have $c \neq x$ and, hence, $\#\{a, b, x, c, d\} = 5$. If $W(bx|dc) > 0$ would not hold, we would either have $W(bc|dx) > 0$ and therefore $W(ab|c|dx)$ implying

$$W(ab|cd) > W(ab|xd) = W(ab|cd) + W(bc|xd) > W(ab|cd),$$

an obvious contradiction, or we would have $W(bd|cx) > 0$ and, hence, also $W(ab|d|cx)$ in contradiction to $W(ab|dc) \neq W(ab|dx)$. Thus, $W(ab|x|dc)$, or equivalently, $W(ab|x|cd)$ must hold, as claimed.

(iv) $\Rightarrow$ **(i)**: We must have $W(ab|xc) > 0$ because, otherwise, we would have either $W(xa|bc) > 0$ and therefore $W(xa|b|cd)$, or $W(xb|ac) > 0$ and therefore $W(xb|a|cd)$, both assertions being in contradiction to our assumption $W(xa|cd) = W(xb|cd)$. By symmetry (exchanging a, b with c, d), we must also have $W(bx|cd) > 0$ implying that $W(ab|x|cd) > 0$ must hold indeed. ∎

Corollary 3.2. *If* $W(ab|cd) > 0$ *and* $W(ab|cd) \geq W(ax|cd), W(bx|cd)$ *hold for any five distinct elements* $a, b, c, d, x \in X$, *one has*

$$W(ab|xc) > 0.$$

Proof. Otherwise, we could assume without loss of generality that $W(xa|bc) > 0$ holds which, together with $W(ab|cd) > 0$, would imply $W(xa|b|cd)$, and hence,

$$W(xa|cd) = W(xa|bc) + W(ab|cd) > W(ab|cd),$$

a contradiction. ∎

Corollary 3.3. *If $W(ab|xy), W(ab|yz) > 0$ holds for any five distinct elements $a, b, x, y, z \in X$, one has*

$$W(ax'|y'z') = W(bx'|y'z')$$

for all $x', y', z' \in X$ with $\{x, y, z\} = \{x', y', z'\}$.

Proof. Our assumptions imply $W(ab|xz) \geq \min\{W(ab|xy), W(ab|yz)\} > 0$. Thus, symmetry (relative to x, y, z) allows us to assume, without loss of generality, that $W(bx|yz) > 0$ holds. Together with $W(ab|xy) > 0$, this implies $W(ab|x|yz)$, and hence,

$$W(ax|yz) = W(bx|yz) > 0,$$

which in turn implies that

$$W(ax'|y'z') = W(bx'|y'z')$$

holds for all x', y', z' with $\{x', y', z'\} = \{x, y, z\}$ because both terms vanish in case $x' \neq x$, and both terms coincide with $W(ax|yz) = W(bx|yz)$ in case $x' = x$. ∎

Corollary 3.4. *If*

$$0 < W(ab|xy) \leq W(ab|xz), W(ab|yz)$$

holds for five distinct elements a, b, x, y, z in X, one has either $W(ab|x|yz)$ or $W(ab|y|xz)$ and, hence, in any case

$$W(ab|xz) = W(ab|xy) + W(ay|xz) = W(ab|xy) + W(by|xz) \tag{3.2}$$

as well as

$$W(ab|yz) = W(ab|xy) + W(ax|yz) = W(ab|xy) + W(bx|yz). \tag{3.3}$$

Proof. Clearly, both $W(ab|x|yz)$ and $W(ab|y|xz)$ imply (3.2) and (3.3). Thus, it is enough to show that either $W(bx|yz) > 0$ or $W(by|xz) > 0$ must hold. Yet, otherwise we would have $W(bz|xy) > 0$ implying that $W(ab|z|xy)$ would hold in contradiction to $W(ab|xy) \leq W(ab|xz)$. ∎

Next, we define

$$\underline{W}(ab|**) := \min\left\{ W(ab|xy) \,\middle|\, \{x,y\} \in \binom{X \setminus \{a,b\}}{2} \right\}$$

for any two distinct elements $a, b \in X$.

Note that in case the map W is of the form $W_{T,\ell}$ for some binary X-tree T and some length function $\ell\colon E_1(T) \to \mathbb{R}_{>0}$, we have $\underline{W}(ab|**) > 0$ for any two distinct vertices a and b if and only if the vertices a and b form a *cherry* in T, i.e., the two unique vertices u, v in V with $\{a, u\}, \{b, v\} \in E$ coincide.

Corollary 3.5. *If*

$$W(a_0b_0|c_0d_0) = \max\left\{W(ab|cd)\,\middle|\, ab|cd \in \mathcal{S}_{2|2}(X)\right\}$$

holds for some $a_0b_0|c_0d_0 \in \mathcal{S}_{2|2}(X)$*, one has* $\underline{W}(a_0b_0|**) > 0$ *as well as* $W(a_0x|yz) = W(b_0x|yz)$ *for all* $\{x,y,z\} \in \binom{X\setminus\{a_0,b_0\}}{3}$.

Proof. Corollary 3.2 implies that $W(a_0b_0|xc_0) > 0$ must hold for all x in $X - \{a_0, b_0, c_0\}$ which in turn implies that $W(a_0b_0|xy) > 0$ holds for all $x, y \in X - \{a_0, b_0\}$ with $x \neq y$, in view of (F4) and, therefore, also

$$W(a_0x|yz) = W(b_0x|yz)$$

for all $\{x,y,z\} \in \binom{X\setminus\{a_0,b_0\}}{3}$ in view of Corollary 3.3. ∎

Corollary 3.6. *If* $0 < W(ab|xy) = \underline{W}(ab|**)$ *holds for four distinct elements* $a, b, x, y \in X$*, one has*

$$W(ab|xz) = W(ab|xy) + W(ay|xz)$$

as well as

$$W(ab|yz) = W(ab|xy) + W(ax|yz)$$

for all $z \in (X \setminus \{a,b,x,y\})$.

Proof. This follows directly from Corollary 3.4. ∎

Next, we define

$$\overline{W}_b(a*|cd) := \max\left\{W(az|cd)\,\middle|\, z \in X \setminus \{a,b,c,d\}\right\}$$

for any four distinct elements $a, b, c, d \in X$. The following result will be crucial for our proof of Theorem 1.1:

Lemma 3.7. *If* $\underline{W}(ab|**) > 0$ *holds for two distinct elements* $a, b \in X$*, one has*

$$W(ab|cd) = \underline{W}(ab|**) + \overline{W}_b(a*|cd) \tag{3.4}$$

for any two distinct elements $c, d \in X \setminus \{a,b\}$*. In particular, a map* W *from* $\mathcal{S}_{2|2}(X)$ *into* $\mathbb{R}_{\geq 0}$ *that satisfies the conditions* (F1) *to* (F4) *is completely determined, for any two distinct elements* $a, b \in X$ *with* $\underline{W}(ab|**) > 0$*, by its values on* $\mathcal{S}_{2|2}(X \setminus a) \cup \mathcal{S}_{2|2}(X \setminus b)$ *and the value of* $\underline{W}(ab|**)$.

Proof. In case $W(ab|cd) = \underline{W}(ab|**)$, we have to show that $W(az|cd) = 0$ holds for all $z \in X \setminus \{a,b,c,d\}$ which follows from the fact that $W(az|cd) > 0$ for some $z \in X \setminus \{a,b,c,d\}$ would imply $W(ba|z|cd)$ in view of $W(ba|zc) > 0$ and $W(az|cd) > 0$ in contradiction to $W(ab|cd) = \underline{W}(ab|**) \leq W(ab|zc)$.

Otherwise, we have $W(ab|cd) > \underline{W}(ab|**)$ and we can use (F4) to find some $z \in X \setminus \{a,b,c,d\}$ with $W(ab|zc) = \underline{W}(ab|**)$ and, therefore, $W(ba|z|cd)$ in view of $W(ab|cd) > W(ab|zc) > 0$ and Lemma 3.1, **(iii)** ⇒ **(i)** ⇒ **(ii)** and, thus,

$$\begin{aligned} W(ab|cd) &= W(ab|cz) + W(az|cd) = \underline{W}(ab|**) + W(az|cd) \\ &\leq \underline{W}(ab|**) + \overline{W}_b(a*|cd). \end{aligned}$$

It remains to show that

$$W(az'|cd) \leq W(az|cd)$$

holds for all $z' \in X \setminus \{a, b, c, d\}$. Otherwise, however, we would have $W(az'|cd) > W(az|cd) > 0$ for some $z' \in X \setminus \{a, b, c, d, z\}$ and, hence, $W(az'|z|cd)$ by Lemma 3.1, **(iii)** $\Rightarrow$ **(i)** $\Rightarrow$ **(ii)** which in turn would imply $W(ba|z'|zc)$ in view of $W(az'|zc) > 0$ and $W(ba|z'z) \geq \underline{W}(ab|**) > 0$, and, hence, $W(ab|z'c) < W(ab|zc)$ in contradiction to $W(ab|zc) = \underline{W}(ab|**) \leq W(ab|z'c)$. ∎

We now turn to the remaining part of the proof of Theorem 1.1. We already showed in the previous section that there can be at most one pair T, ℓ with $W = W_{T,\ell}$. So, it remains to show that such an X-tree T and a length function ℓ indeed exist.

To this end, we will use induction relative to the cardinality n of X. Clearly, Theorem 1.1 holds in case $n = 4$. Indeed, if the elements in X are labelled a, b, c, d so that $W(ab|cd) > 0$ and, hence, $W(ac|bd) = W(ad|bc) = 0$ holds, the tree

$$\begin{aligned} T &= T_{ab|cd} \\ &:= \Big(\{a, b, c, d, u_{ab}, u_{cd}\}, \Big\{\{a, u_{ab}\}, \{b, u_{ab}\}, \{c, u_{cd}\}, \{d, u_{cd}\}, \{u_{ab}, u_{cd}\}\Big\}\Big) \end{aligned}$$

with exactly four leaves a, b, c, d and two additional vertices named u_{ab}, u_{cd} of degree 3, u_{ab} adjacent to a, b, and u_{cd}, u_{cd} adjacent to c, d, and u_{ab}, together with the map

$$\ell\colon \big\{\{u_{ab}, u_{cd}\}\big\} \to \mathbb{R}_{>0}, \quad \{u_{ab}, u_{cd}\} \mapsto W(ab|cd)$$

is obviously the unique required pair T, ℓ with $W = W_{T,\ell}$.

To perform induction, we now assume $n > 4$ and choose $a_0b_0\big|c_0d_0 \in \mathcal{S}_{2|2}(X)$ with

$$W(a_0b_0\big|c_0d_0) \geq W(ab\big|cd) \tag{3.5}$$

for all $ab\big|cd \in \mathcal{S}_{2|2}(X)$.

In view Corollary 3.5, this implies that $\underline{W}(a_0b_0|**) > 0$ as well as

$$W(a_0x\big|yz) = W(b_0x\big|yz) \tag{3.6}$$

for any three distinct elements $\{x, y, z\}$ in $X - \{a_0, b_0\}$.

Next, using our inductive hypothesis, we choose a binary $(X \setminus \{a_0\})$-tree T_1 and a length function $\ell_1\colon E_1(T_1) \to \mathbb{R}_{>0}$ with

$$W_{T_1,\ell_1} = W\big|_{\mathcal{S}_{2,2}(X-\{a_0\})}$$

and note that, in view of (3.6), we have also

$$W_{T_2,\ell_2} = W\big|_{\mathcal{S}_{2,2}(X-\{b_0\})},$$

for the binary $(X - \{b_0\})$-tree T_2 and the length function $\ell_2\colon E_1(T_2) \to \mathbb{R}_{>0}$ derived by renaming the vertex a_0 in T_1 by b_0.

Let u_0 denote the unique vertex in $V(T_1)$ with $\{u_0, b_0\} \in E(T_1)$ (and, hence, with $\{u_0, a_0\} \in E(T_2)$). It is clear that u_0 is not a leaf in either T_1 or T_2. Now, choose

some further element w_0 not in any set previously considered and define $T = (V, E)$ and $\ell\colon E_1(T) \to \mathbb{R}_{>0}$ as follows:

$$V := V(T_1) \cup \{a_0, w_0\},$$

$$E := \big\{\{a_0, w_0\}, \{b_0, w_0\}\{u_0, w_0\}\big\} \cup E(T_1) \setminus \big\{\{b_0, u_0\}\big\}.$$

Note that

$$E_1(T) = E_1(T_1) \cup \big\{\{u_0, w_0\}\big\}$$

holds. Put

$$\ell(e) = \ell_1(e)$$

for all $e \in E_1(T_1)$, and

$$\ell(\{u_0, w_0\}) := \underline{W}(a_0 b_0 | * *). \tag{3.7}$$

One has to show that $W = W_{(T,\ell)}$ holds. However, both maps coincide on $\mathcal{S}_{2|2}(X \setminus a_0) \cup \mathcal{S}_{2|2}(X \setminus b_0)$ in view of our construction, and we have also $\underline{W}_{(T,\ell)}(a_0 b_0 | * *) = \ell(\{u_0, w_0\}) = \underline{W}(a_0 b_0 | * *)$. Thus, our claim follows from Lemma 3.7. ∎

The observations leading to this proof immediately suggest various algorithms to construct the tree and to determine the length function: First one has to determine a suitable labelling $X = \{a_1, a_2, \ldots, a_n\}$ of the elements in X and then, in a second run, one builds the tree in a recursive fashion.

4. Discussion

The crucial observation used above that a map $W\colon \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$ which satisfies the conditions (F1)–(F4) and certain inequalities is uniquely determined by its restriction to a certain subset of $\mathcal{S}_{2|2}(X)$, raises the question for which other collections of inequalities and corresponding subsets of $\mathcal{S}_{2|2}(X)$ this might hold. E.g., one can generalize the observation above and show that, given any four distinct elements a_1, a_2, a_3, a_4 in X with

$$0 < W(a_1 a_2 | a_3 a_4) \leq W(a_1' a_2' | a_3' a_4')$$

for all $\{a_1', a_2', a_3', a_4'\} \in \binom{X}{4}$ with $W(a_1' a_2' | a_3' a_4') > 0$ and

$$\#(\{a_1, a_2, a_3, a_4\} \cap \{a_1', a_2', a_3', a_4'\}) = 3,$$

the map W is uniquely determined by its restriction to all 4-subsets $\{x_1, x_2, x_3, x_4\}$ of X for which $\{x_1, x_2, x_3, x_4\}$ is either contained in

$$A_1 := \{a_1, a_2, a_3\} \cup \Big\{a \in X \setminus \{a_1, a_2, a_3\} \Big| W(a_1 a | a_2 a_3) > 0\Big\},$$

or in

$$A_2 := \{a_1, a_2, a_3\} \cup \Big\{a \in X \setminus \{a_1, a_2, a_3\} \Big| W(a a_2 | a_1 a_3) > 0\Big\},$$

or in

$$A_3 := \{a_1, a_3, a_4\} \cup \Big\{a \in X \setminus \{a_1, a_3, a_4\} \Big| W(a_1 a_4 | a a_3) > 0\Big\},$$

or, finally, in

$$A_4 := \{a_1, a_3, a_4\} \cup \left\{ a \in X \setminus \{a_1, a_3, a_4\} \middle| W(a_1a_3|aa_4) > 0 \right\}.$$

Using this observation, the required X-tree T and length function ℓ with $W = W_{T,\ell}$ can also be constructed as follows: One first chooses two distinct elements a_1, a_2 in X for which some subset $\{x, y\} \in \binom{X \setminus \{a_1, a_2\}}{2}$ with $W(a_1a_2|xy) > 0$ exists, then one chooses two distinct elements a_3, a_4 in $X \setminus \{a_1, a_2\}$ with

$$W(a_1a_2|a_3a_4) = \min \left\{ W(a_1a_2|xy) \middle| \{x, y\} \in \binom{X \setminus \{a_1, a_2\}}{2}, W(a_1a_2|xy) > 0 \right\},$$

and observes that $W(a_1a_2|a_3a_4) \leq W(a'_1a'_2|a'_3a'_4)$ must hold for all $\{a'_1, a'_2, a'_3, a'_4\} \in \binom{X}{4}$ with $W(a'_1a'_2|a'_3a'_4) > 0$ and $\#(\{a_1, a_2, a_3, a_4\} \cap \{a'_1, a'_2, a'_3, a'_4\}) = 3$, then one constructs the subsets A_1, A_2, A_3, A_4 as above and, noting that $a_4 \notin A_1 \cup A_2$ and $a_2 \notin A_3 \cup A_4$ hold, and then one uses the induction hypothesis to find, for each $i \in \{1, 2, 3, 4\}$, an A_i-tree T_i together with a length function ℓ_i such that $W_{T_i,\ell_i} = W|_{\mathcal{S}_{2|2}(A_i)}$ holds. Finally, one "fuses" these four "small" trees in an appropriate (and absolutely canonical) way into one big supertree T and one uses the length function $\ell_1, \ell_2, \ell_3, \ell_4$ to define a length function ℓ for T for which one finally observes that $W = W_{T,\ell}$ must hold by referring to the above generalization of Corollary 3.5.

More generally, one may as well start with any arbitrary labelling

$$X = \{a_1, a_2, \ldots, a_n\}$$

of the elements in X and use the above analysis to construct recursively, starting with the tree $T_3 := (\{a_1, a_2, a_3, v\}, \{\{a_i, v\} | i = 1, 2, 3\})$, a sequence of trees $T^{(i)}$ with leave set $X_i := \{a_1, \ldots, a_i\}$ and length function ℓ_i defined on $E_1(T_i)$ for $i = 4, \ldots, n$ such that

$$W\Big|_{\mathcal{S}_{2|2}(X_i)} = W_{T_i,\ell_i}$$

holds for all $i = 4, \ldots, n$.

Indeed, comparing W-values, one can — for each $i = 4, \ldots, n$ — identify that edge $e_i = \{u_i, v_i\}$ in $T^{(i-1)}$ to which the new pending edge with leaf a_i has to be attached. The tree $T^{(i)}$ then results from $T^{(i-1)}$ by eliminating the edge e_i and adding a new internal vertex w_i as well as three new edges $\{u_i, w_i\}, \{w_i, v_i\}, \{w_i, a_i\}$, and the length function ℓ_i can then also be defined easily on the (one or two) new internal edges while keeping the value of ℓ_{i-1} on all internal edges of $T^{(i)}$ that are also internal edges of $T^{(i-1)}$.

While, given a map W that satisfies the conditions (F1) to (F4), the outcome of any such recursive construction does, of course, not depend on the labelling of X, the algorithmic procedure will selective only use certain W-values (depending strongly on the chosen labelling) and can thus be applied to any map W from $\mathcal{S}_{2|2}(X)$ into $\mathbb{R}_{\geq 0}$ whether or not (F1) to (F4) are satisfied. And it will always produce a weighted X-tree depending on that map W and the input labelling.

In a forthcoming paper, we will discuss various ideas on how to make a sensible choice of the input labelling in case one starts with a map W that satisfies the conditions (F1) to (F4) only approximately, and present some related experimental results.

Our result also suggests to study arbitrary subsets $\mathcal{X}$ of $\mathcal{S}_{2|2}(X)$ and maps $W_0 : \mathcal{X} \to \mathbb{R}_{\geq 0}$ and ask for necessary and/or sufficient conditions on $\mathcal{X}$ and W_0 that imply that there exists at least (or at most) one extension $W = \mathcal{S}_{2|2}(X) \to \mathbb{R}_{>0}$ of W_0 that satisfies the conditions ($\mathcal{X}$) as well as perhaps certain inequalities, or for algorithms that decide extendability and/or construct such an extension if it exists. The results by Boecker and others (cf. [2–4]) suggest that deciding unique extendability might, at least in certain cases, be considerably simpler than just deciding extendability.

Another question that arises naturally in this context is how, given any map $W : \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$, one can find a map $W' : \mathcal{S}_{2|2}(X) \to \mathbb{R}_{\geq 0}$ that satisfies the conditions (F1)–(F4) and approximates W as closely as possible (relative to some predefined measure of "closeness"). While prescribing the support of W' (i.e., the topology of the X-tree in question), least square approximations should be easy, a linear-programming approach (similar to that pursued by Weyer-Menkhoff [40], see also [24]) in the case of unweighted X-trees where only the support of W' is of interest) would be welcome whenever any a priori assumptions about that support cannot be provided.

References

1. H.-J. Bandelt and A. Dress, Reconstructing the shape of a tree from observed dissimilarity data, Adv. Appl. Math. **7** (1986) 309–343.
2. S. Böcker, From subtrees to supertrees, Ph.D. Thesis, Universität Bielefeld, 1999, pp. 1–100.
3. S. Böcker, A.W.M. Dress, and M.A. Steel, Patching up X-trees, Ann. Combin. **3** (1999) 1–12.
4. S. Böcker, D. Bryant, A.W.M. Dress, and M.A. Steel, Algorithmic aspects of tree amalgamation, J. Algorithm **37** (2000) 522–537.
5. H. Colonius and H.H. Schultze, Trees constructed from empirical relations, Braunschweiger Berichte aus dem Institut fuer Psychologie **1** (1977).
6. H. Colonius and H.H. Schultze, Tree structure for proximity data, British J. Math. Statist. Psych. **34** (1981) 167–180.
7. J.H. Badger and P. Kearney, Picking fruit from the tree of life, In: Proc. 16th ACM Symp. Appl. Comput., Las Vegas, March 11–14, 2001, pp. 61–67.
8. A. Ben-Dor, B. Chor, D. Graur, R. Ophir, and D. Pelleg, Constructing phylogenies from quartets: elucidation of Eutherian superordinal relationships, J. Comput. Biol. **5** (3) (1998) 377–390.
9. V. Berry, T. Jiang, P. Kearney, M. Li, and T. Wareham, Quartet cleaning: improved algorithms and simulations, In: Algorithms — ESA'99, 7th European Symposium on Algorithms Prague, Chezh Rep. Lect. Notes Comput. Sci., Vol. 1643, 1999, pp. 313–324.
10. V. Berry and O. Gascuel, Inferring evolutionary trees with strong combinatorial evidence, Theoret. Comput. Sci. **240** (2000) 271–298.
11. V. Berry, D. Bryant, T. Jiang, P. Kearney, M. Li, T. Wareham, and H. Zhang, A practical algorithm for recovering the best supported edges of an evolutionary tree (extended abstract), In: ACM Symp. on Discrete Algorithms SODA2000, 2000, pp. 287–296.
12. O. Bininda-Emonds, S.G. Brady, J. Kim, and M.J. Sanderson, Scaling of accuracy in extremely large phylogenetic trees, In: 6th Pacific Symp. on Biocomputing, 2001, pp. 547–558.
13. D.J. Bryant and M.A. Steel, Extension operations on sets of leaf-labelled trees, Adv. Appl. Math. **16** (1995) 425–453.

14. D. Bryant and M. Steel, Fast algorithms for constructing optimal trees from quartets, In: Proc. Tenth Annual ACM-SIAM Symposium on Discrete Algorithms, Baltimore, Maryland, 1999, pp. 147–155.
15. P. Buneman, The recovery of trees from measures of dissimilarity, In: Mathematics in the Archaeological and Historical Sciences, F.R. Hodson, D.G. Kendall, and P. Tautu, Eds., Edinburgh University Press, Edinburgh, 1971, pp. 387–395.
16. B. Chor, Form quartets to phylogenetic trees, In: SOFSEM'98: Theory and Practice of Informatics, B. Rovan, Ed., Lecture Notes in Computer Science, Vol. 1521, Springer-Verlag, 1998, pp. 36–53.
17. M. Csűrös and M-Y. Kao, Provable and accurate recovery of evolutionary trees through harmonic greedy triplets, SIAM J. Comput. **31** (2001) 306–322.
18. M. Csűrös, Fast recovery of evolutionary trees with thousands of nodes, J. Comput. Biol. **9** (2002) 277–297.
19. M.C.H. Dekker, Reconstruction methods for derivation trees, Master's Thesis, Vrije Universiteit, Amsterdam, 1986.
20. A. Dress, M. Hendy, K. Huber, and V. Moulton, Enumerating the vertices of the Buneman graph, Preprints Forschungsschwerpunkt Mathematisierung/Strukturbildungsprozesse, **117**, 1997.
21. P.L. Erdős, M.A. Steel, L.A. Székely, and T. Warnow, Local quartet splits of a binary tree infer all quartet splits via one dyadic inference rule, Comput. Artificial Intelligence **16** (2) (1997) 217–227.
22. P.L. Erdős, M.A. Steel, L.A. Székely, and T. Warnow, Inferring big trees from short quartets, In: Automata, Languages and Programming 24th International Colloquium, ICALP'97, Bologna, Italy, July 7–11, 1997, P. Degano, R. Gorrieri, A. Marchetti-Spaccamela, Eds., Lecture Notes in Computer Science, Vol. 1256, 1997, pp. 827–837.
23. J. Gramm and R. Niedermeier, Minimum quartet inconsistency is fixed parameter tractable, In: Combinatorial Pattern Matching, CPM2001, A. Amir and G.M. Landau Eds., Israel, Jerusalem, LNCS 2089, 2001, pp. 241–256.
24. S. Grünewald, The quartet joining algorithm, manuscript, Bielefeld, 2002.
25. D. Huson, S. Nettles, L. Parida, T. Warnow, and S. Yooseph, The disk-covering method for tree reconstruction, In: Proceedings of "Algorithms and Experiments," ALEX'98, Trento, Italy, 1998, pp. 62–75.
26. D. Huson, S. Nettles, K. Rice, T. Warnow, and S. Yooseph, Hybrid tree reconstruction methods, ACM J. Exp. Alg. **4** (1998) Article 5.
27. D.H. Huson, S.M. Nettles, and T.J. Warnow, Disk-covering, a fast-converging method for phylogenetic tree reconstruction, J. Comput. Biol. **6** (3/4) (1999) 369–386.
28. T. Jiang, P. Kearney, and M. Li, Orchestrating quartets: approximation and data correction, FOCS'98 Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science, 1998, pp. 416–425.
29. T. Jiang, P. Kearney, and M. Li, A polynomial time approximation scheme for inferring evolutionary trees from quartet topologies and its application, SIAM J. Comput. **30** (2000) 1942–1961.
30. P.E. Kearney, The ordinal quartet method (extended abstract), In: RECOMB'98, New York, 1998, pp. 125–133.
31. J. Kim, large-scale phylogenies and measuring the performance of phylogenetic estimators, Syst. Biol. **47** (1998) 43–60.
32. J. Lagergren, Combining polynomial running time and fast convergence for the disk-covering method, J. Comput. System Sci. **65** (2002) 481–493.

33. L. Nakhleh, U. Roshan, K.St. John, J. Sun, and T. Warnow, Designing fast converging phylogenetic methods, In: Bioinformatics, Oxford University Press, ISMB'01 **17** (90001), 2001, S190–S198.
34. V. Ranwez and O. Gascuel, Quartet based phylogenetic inference: improvements and limits, Mol. Biol. Evol. **18** (6) (2001) 1103–1116.
35. K. Strimmer and A. von Haeseler, Quartet puzzling: a quartet maximum likelihood method for reconstructing tree topologies, Mol. Biol. Evol. **13** (1996) 964–969.
36. K. Strimmer, N. Goldman, and A. von Haeseler, Bayesian probabilities and quartet puzzling, Mol. Biol. Evol. **14** (1997) 210–211.
37. M.A. Steel, L.A. Székely, and P.L. Erdős, The number of nucleotide sites needed to accurately reconstruct large evolutionary trees, DIMACS Technical Report 1996–19.
38. G.D. Vedova and H.T. Wareham, Optimal algorithms for local vertex quartet cleaning, Bioinformatics **18** (2002) 1297–1304.
39. T. Warnow, B.M.E. Moret, and K.St. John, Absolute convergence: true trees from short sequences, In: ACM Symp. on Discrete Algorithms SODA'01, 2001, pp. 186–195.
40. J. Weyer-Menkhoff, Phylogenetic Combinatorics, Ph.D. Thesis, Bielefeld, 2003.

Annals of Combinatorics 10 (2006) 415-430
0218-0006/06/040415-16
DOI 10.1007/s00026-006-0297-3

Annals of Combinatorics

Subwords in Reverse-Complement Order*

Péter L. Erdős[1], Péter Ligeti[2], Péter Sziklai[2], and David C. Torney[3]

[1]A. Rényi Institute of Mathematics, Hungarian Academy of Sciences, Budapest,
P.O. Box 127, H-1364, Hungary
elp@renyi.hu

[2]Department of Computer Science, Eötvös University, Pázmány Péter sétány 1/C,
H-1117 Budapest, Hungary
{turul, sziklai}@cs.elte.hu

[3]Theoretical Biology and Biophysics, Mailstop K710, Los Alamos National Laboratory,
Los Alamos, New Mexico, 87545, USA
dtorney@earthlink.net

Received October 19, 2005

AMS Subject Classification: 05D05, 68R15

Abstract. We examine finite words over an alphabet $\Gamma = \{a, \bar{a}; b, \bar{b}\}$ of pairs of letters, where each word $w_1 w_2 \cdots w_t$ is identified with its *reverse complement* $\bar{w}_t \cdots \bar{w}_2 \bar{w}_1$ $\left(\text{where } \bar{\bar{a}} = a, \bar{\bar{b}} = b\right)$. We seek the smallest k such that every word of length n, composed from Γ, is uniquely determined by the set of its subwords of length up to k. Our almost sharp result ($k \sim 2n/3$) is an analogue of a classical result for "normal" words. This problem has its roots in bioinformatics.

Keywords: combinatorics of words, Levenshtein distance, DNA codes, reconstruction of words

1. Introduction

Let Δ be a finite alphabet and let Δ^* denote the set of all finite sequences over Δ, called *words*. For $s, w \in \Delta^*$ we say that s is a *subword* of w ($s \leq w$) if s is a (not necessarily consecutive) subsequence of w. (Note, that some authors have called these constructs "subsequences", reserving "subword" for consecutive subsequences.) The length of w is denoted by $|w|$. The following result was independently rediscovered repeatedly; as far as we are aware the problem originally was posed by Schützenberger and Simon. (In the bibliography we try to give the original sources relevant to our problem. It is not our intention, however, to give a comprehensive bibliography.)

Theorem 1.1. (Simon [8]) *Every word $w \in \Delta^*$ with at most $2m-1$ letters is completely determined by its length and by the set of all its subwords of length at most m.*

* This work was supported, in part, by Hungarian NSF, under contract Nos. AT48826, NK62321, F043772, N34040, T34702, T37846, T43758, ETIK, Magyary Z. grant and by the U.S.D.O.E..

The pair of words *abababa* and *bababab* shows clearly that this result is sharp. In Simon's paper it was noted that it suffices to prove the theorem for the two-letter case: $\Delta = \{a, b\}$. Perhaps the shortest proof of Theorem 1.1 is due to Sakarovitch and Simon (see [6, pp. 119–120]); we were influenced by this nice proof.

Levenshtein in his papers [3–5] considers more generalizations of the reconstruction problem. In [3] the author examines which other sets of subwords or super-words determine uniquely the original word, in [4] the maximum size of the set of common subwords (or super-words) of two different words of a given length is given in a recursive way. In [5] every unknown sequence is reconstructed from its versions distorted by errors of a certain type, which are considered as outputs of repeated transmissions over a channel, and a minimal number of transmissions sufficient to reconstruct the original word (either exactly or with a given probability) is given. In both of the latter papers simple reconstruction algorithms are given.

In this paper we study another version of this problem. Let $\Gamma = \{a, \bar{a}; b, \bar{b}\}$ be an alphabet where the letters come in pairs (called *complement pairs*); and let Γ^* denote the set of all finite sequences, called *words*, composed from Γ. Define $\bar{\bar{a}} = a$, $\bar{\bar{b}} = b$ and for a word $w = w_1 w_2 \cdots w_t \in \Gamma^*$ let $\widetilde{w} = \overline{w_t}\,\overline{w_{t-1}} \cdots \overline{w_1}$, the *reverse complement* of w. Note that $\widetilde{(\widetilde{w})} = w$. Now we want to keep the essence of the previous partial ordering, while, in our poset, *each word is identified with its reverse complement.*

As in the foregoing theorem, we do not address effective *reconstruction* essentially; our concern is the prefatory problem of determining the minimal m such that the subwords of length up to m determine each word of length n. In the "classical case" the reconstruction problem was recently addressed (see, Dress and Erdős [1]). In the reverse complement case the problem seems to be more complicated, and no results are presently available.

Our problem and definitions have biological motivations (for details see [2]). DNA typically exists as paired, reverse complementary words or *strands*: The Watson-Crick double helix, with its four letters, A, C, G and T paired via $\bar{A} = T$ and $\bar{C} = G$. Corresponding DNA codes could involve the insertion-deletion metric — with bounded *similarity* between two strands: The length of the longest subword common either to the strands or common to one strand and the reverse complement of the other.

Another common task is to decide rapidly and efficiently whether a given DNA double-strand (for example an erroneous gene, which is associated with illness) is present in a sample. This setting typically invokes microarrays: Ten thousand or so of relatively short DNA words (called *probes*) are fixed on a glass slide. The sample reacts with the probes, and the probes which bind material from the sample are determined. We may model this process with our definition, i.e., to say that binding occurs if the probe is a subword of either strand. One may argue that the physicochemical laws do not allow each subword of the long DNA word to bind effectively because, for instance, "blocks" of consecutive matches may be required for binding. Although this is a perfectly legitimate objection, our aim is to provide additional background for such applications.

Before we list our main results, let us remark that our problem is a special case of a general class of problems, in which group orbits substitute for the classes of words and their reverse complements. The group must have a well defined action on all subwords

— an induced action based, for instance, on permuting letter identities and letter positions. (The group considered herein is of order two.) A permutation may, for example, act on the positions included in subwords through the respective complete ordering. Thus, one version of the general problem is:

Given the k-spectra of the words for its orbits (the set of subwords of up to length k occurring in any of these words), find a characterization of all the (permutation) groups which yield k-spectra one-to-one correspondence with these orbits. For the general problem, the respective partial order would be inclusion when any member of the orbit occurs as a subword.

2. Main Results

In this section we formulate our main results. Let us recall that in our partial order every word is identified with its reverse complement. Therefore, if in this partial order the word g is smaller than the word f, then it can happen that g is a subword of f or it is a subword of its reverse complement $\widetilde{f}$. For convenience, if we do not know (or do not care) which is the case, then we will say that the word g *precedes* the word f ($g \prec f$). Let $S(m, f)$ denote the set of words of length $\leq m$, which precede f. We seek to determine when $S(m, f)$ uniquely defines f.

One may note essential differences between this and the original problem; here, for instance, we may have more subwords but we do not distinguish between individual subwords belonging to a word or to its reverse complement. This difference is evident when the alphabet consists of a letter and its complement.

Let us consider the following example:

$$\mathcal{F}' = \bar{a}^{2k+\varepsilon} a^k \quad \text{and} \quad \mathcal{G}' = \bar{a}^{2k+\varepsilon-1} a^{k+1}, \tag{2.1}$$

where $\varepsilon \in \{0, 1, 2\}$, $k \geq 1$ and $(k, \varepsilon) \neq (1, 0)$. The length of both words is $3k+\varepsilon$. On the one hand, the subword $\bar{a}^{2k+\varepsilon}$ of $\mathcal{F}'$ satisfies $\bar{a}^{2k+\varepsilon} \not\prec \mathcal{G}'$. On the other hand, it is easy to check that

$$S\left(2k+\varepsilon-1, \mathcal{F}'\right) = S\left(2k+\varepsilon-1, \mathcal{G}'\right).$$

In this paper we prove the following result:

Theorem 2.1. *Every word $f \in \{a, \bar{a}\}^*$ of length at most $3m-1$ is uniquely determined by its length and by the set*

$$D'(f) := S(2m, f).$$

The proof of this result can be found in Section 4.

The next example illustrates that if our words contain letters from more than one complement pair, then they are "easier to distinguish". Consider the following words:

$$\mathcal{F} = \bar{a}^{2k+\varepsilon} \bar{b}\, b\, a^k \quad \text{and} \quad \mathcal{G} = \bar{a}^{2k+\varepsilon-1} \bar{b}\, b\, a^{k+1}, \tag{2.2}$$

where $\varepsilon \in \{0, 1, 2\}$ and $k \geq 1$ and $(k, \varepsilon) \neq (1, 0)$. The length of both words is $3k+2+\varepsilon$. On the one hand, the subword $\bar{a}^{2k+\varepsilon}$ of $\mathcal{F}$ satisfies $\bar{a}^{2k+\varepsilon} \not\prec \mathcal{G}$. On the other hand, it is easy to verify that

$$S(2k+\varepsilon-1, \mathcal{F}) = S(2k+\varepsilon-1, \mathcal{G}).$$

We have the following statement:

Theorem 2.2. *Every word $f \in \Gamma^*$ of length at most $3m+1$ $(m > 1)$ containing both $(a$ or $\bar{a})$ and $\left(b \text{ or } \bar{b}\right)$ is uniquely determined by its length and by the set*

$$D(f) := S(2m, f).$$

The examples *abab* and *abba* show that in case of $m = 1$ the statement is not true. The proof of this result can be found in Section 5.

Please recognize that due to our definitions, the expression "uniquely determined" means "uniquely determined, up to reverse complementation". The statement pertains to the case of $\varepsilon = 2$ in the example.

3. Easy Consequences

There are some immediate consequences of the results of Section 2. For example in the case when our words contain letters from one complement pair only, one may formulate the following result.

Corollary 3.1. *Every word $f \in \{a, \bar{a}\}^*$ of length at most n is uniquely determined by its length and by the set $S\left(\left\lceil \frac{2(n+2)}{3} \right\rceil, f\right)$.*

Proof. Let m be the smallest integer such that $n \leq 3m-1$. Then $\left\lceil \frac{2(n+2)}{3} \right\rceil \geq 2m$ and Theorem 2.1 applies. ∎

Correspondingly, for the case of words containing letters from two complement pairs, we have

Corollary 3.2. *Every word $f \in \Gamma^*$ of length at most n containing both $(a$ or $\bar{a})$ and $(b$ or $\bar{b})$ is uniquely determined by its length and by the set $S\left(\left\lfloor \frac{2(n+1)}{3} \right\rfloor, f\right)$.*

Proof. The statement is straightforward: Let m be the smallest integer such that $n \leq 3m+1$. Then $\left\lfloor \frac{2(n+1)}{3} \right\rfloor \geq 2m$, therefore Theorem 2.2 applies. ∎

Our instinct says that Corollaries 3.1 and 3.2 are not sharp. We suspect that the truth is the following:

Conjecture 3.3. *Each word of length at most $3m+2+\varepsilon$ containing both $(a$ or $\bar{a})$ and $(b$ or $\bar{b})$ is uniquely determined by its length and by the set $S(2m+\varepsilon, f)$. Furthermore, each word of length at most $3m+\varepsilon$ containing only a or $\bar{a}$ is uniquely determined by its length and by the set $S(2m+\varepsilon, f)$.*

If our words are self-reverse complementary, then we are back to the original problem:

Remark 3.4. Let the words f and $g \in \Gamma^*$ (of length at most n) be self-reverse complementary, that is $f = \tilde{f}$ and $g = \tilde{g}$. Now if $S(\lceil (n+1)/2 \rceil, f) = S(\lceil (n+1)/2 \rceil, g)$ then $f = g$.

Proof. If for the word w we have $w \prec f$ and $f = \tilde{f}$, then w is a subword of f as well as of $\tilde{f}$. Therefore Theorem 1.1 applies. ∎

For the original problem it was almost trivial that from the result for the case of 2-letter alphabet one derives an (approximate) result for the case of k-element alphabets as well. The situation here is similar but the proof requires some work:

Theorem 3.5. *Theorem 2.2 remains valid if the word f contains letters from $k \geq 2$ different complement pairs.*

Proof. We use induction on the number k of different complement pairs present. The case of two pairs present is Theorem 2.2. Assume that the statement is valid for the case of $k-1$ different pairs present. Let f and g be words with length $|f| = |g| \leq 3m+1$, and in both words let there be k different complement pairs present. The alphabet is $\{a_1, \bar{a}_1, \ldots, a_k, \bar{a}_k\}$. Let $A_{1,2}, \bar{A}_{1,2}$ be a new pair of complementary letters, and $f_{1,2}$ be the word derived from f by identifying all occurrences of a_1 and a_2 with $A_{1,2}$ and all occurrences of $\bar{a}_1$ and $\bar{a}_2$ with $\bar{A}_{1,2}$. The word $g_{1,2}$ is derived similarly. The new words contain letters from $k-1$ different pairs and $D(f_{1,2}) = D(g_{1,2})$. The inductive hypothesis gives that $f_{1,2} = g_{1,2}$ $\big($one might need to exchange the names of $g_{1,2}$ and $\widetilde{g}_{1,2}\big)$. Furthermore, for the subwords $f^*_{1,2}$ and $g^*_{1,2}$ consisting of all occurrences of the letters $\{a_1, \bar{a}_1, a_2, \bar{a}_2\}$ we have $D\big(f^*_{1,2}\big) = D\big(g^*_{1,2}\big)$; therefore, we can apply Theorem 2.2. Whence $f^*_{1,2} = g^*_{1,2}$ or $f^*_{1,2} = \widetilde{g}^*_{1,2}$.

In the case of $f^*_{1,2} = g^*_{1,2}$ interleaving $f_{1,2}$ and $f^*_{1,2}$ we can determine f which is identical to g. In case of $f_{1,2} = \widetilde{g}_{1,2}$ and $f^*_{1,2} = \widetilde{g}^*_{1,2}$ we can proceed similarly. However, it can happen that

$$f_{1,2} = g_{1,2}, \text{ but } \quad f_{1,2} \neq \widetilde{g}_{1,2}, \text{ while} \tag{3.1}$$

$$f^*_{1,2} \neq g^*_{1,2}, \text{ but } \quad f^*_{1,2} = \widetilde{g}^*_{1,2}. \tag{3.2}$$

The value $\big|f^*_{1,2}\big|$ cannot be odd, since otherwise $f_{1,2}\left(\frac{|f^*_{1,2}|+1}{2}\right) = g_{1,2}\left(\frac{|g^*_{1,2}|+1}{2}\right)$, therefore $f^*_{1,2} = \widetilde{g}^*_{1,2}$ cannot occur. So let $\big|f^*_{1,2}\big| = \ell$ be even. From Condition (3.2) it follows that there is an index $j \leq \ell/2$ such that $f^*_{1,2}(j) = a_1$, $g^*_{1,2}(j) = a_2$, while $f^*_{1,2}(\ell+1-j) = \bar{a}_2$ and $g^*_{1,2}(\ell+1-j) = \bar{a}_1$. From Condition (3.1) it follows that there is a subscript $i \leq (3m+1)/2$ such that $f_{1,2}(i) = a_3$ (therefore $g_{1,2}(i) = a_3$ also holds) while $g_{1,2}(3m+2-i) = b$ where $b \neq \bar{a}_3$. If $b \in \{a_1, \ldots, a_k\}$, then introducing the new letters $B_1, \bar{B}_1, B_2, \bar{B}_2$, substitute all occurrences of a_1 and a_3 with B_1, all occurrences of $\bar{a}_1, \bar{a}_3$ with $\bar{B}_1$, all occurrences of the letters $a_2, a_4, \ldots, a_k$ with B_2, and, finally, all occurrences of the letters $\bar{a}_2, \bar{a}_4, \ldots, \bar{a}_k$ with $\bar{B}_2$ in the original words. The result is the words f^B and g^B which satisfy the conditions of Theorem 2.2 while clearly $f^B \neq g^B$ and $f^B \neq \widetilde{g^B}$, a contradiction.

If, however, $b \in \{\bar{a}_1, \bar{a}_2, \bar{a}_4, \ldots, \bar{a}_k\}$, then we may define a bipartition of the alphabet, where letters b and a_3 belong to different classes, and letters a_1 and a_2 also belong to different classes. Then substitute all occurrences of the letters from the first class of the bipartition with $C_1, \bar{C}_1$ and the letters from the second class with $C_2, \bar{C}_2$, respectively. The new words clearly satisfy the conditions of Theorem 2.2; however, the consequence of Theorem 2.2 does not hold. ∎

This proof suggests that the existence of letters from more complement pairs decreases the necessary subword length in the result.

Because our approach does not work for very short words, we use the following enumerative result:

Remark 3.6. Theorems 2.1 and 2.2 were tested by a computer program for short words (for $|f| \leq 13$ and for selected words with $|f| \leq 18$) and were found valid. Therefore our proofs need only address sufficiently long words, allowing reasoning which is effective above a (usually very small) length.

In the next two sections we prove our main results. The general approach used is similar to the one in the proof of Theorem 3.5: Identify a subword of the word under investigation which distinguishes the word and its reverse complement from each other. Such a subword can identify the word itself. The greater the similarity between the word and its reverse complement, the harder to find such a subword but, compensating for this difficulty, the more is known about the structure of such words.

4. The Proof of Theorem 2.1

Assume that f and g are words in $\{a, \bar{a}\}^*$ of the same length such that

$$|f| = |g| \leq 3m - 1 \quad \text{and} \quad D'(f) = D'(g) = D'.$$

Due to Remark 3.4, we may assume that f is not self-reverse complementary. Denote by $A(w)$ the number of a's in the word w, and define $\bar{A}(w)$ analogously. Without loss of generality we may assume that both words f and g are written in the form where $A(f) \geq \bar{A}(f)$ and $A(g) \geq \bar{A}(g)$. At first assume that $A(f) > A(g)$, which also means that $\bar{A}(f) < \bar{A}(g)$. If $A(f) > 2m$, then take an arbitrary subword g' of g such that $A(g'), \bar{A}(g') \geq \bar{A}(f) + 1$. It is clear that $g' \not\sim f$. If, instead, $A(f) \leq 2m$, then take the subword f' of f containing $A(g) + 1$ a's. It is also clear that $f' \not\sim g$ and that $|f'|, |g'| \leq 2m$, which constitutes a contradiction. Therefore, in this proof henceforth we assume that we have

$$A := A(f) = A(g) \quad \text{and} \quad \bar{A} := \bar{A}(f) = \bar{A}(g). \tag{4.1}$$

Before proceeding we introduce one more notion: a word contains a *run* of length k when it contains k consecutive copies of a certain letter.

4.1. The Case $\bar{A} < A$

In this case we know that $f \neq \tilde{f}$ and $g \neq \tilde{g}$, and each subword of f or g containing at least $\bar{A} + 1$ a's obeys these inequalities. All subwords from $S(2m, f)$, containing at least $\bar{A} + 1$ a's, are subwords of g, because they cannot be subwords of $\tilde{g}$ — and correspondingly, the analogous statement holds for the subwords from $S(2m, g)$.

Our words f and g can be written in the following form:

$$f := a^{I_0}\bar{a}\,a^{I_1}\bar{a}\cdots\bar{a}\,a^{I_s} \quad \text{and} \quad g := a^{J_0}\bar{a}\,a^{J_1}\bar{a}\cdots\bar{a}\,a^{J_s},$$

where $s = \bar{A}$, and any I_l or J_l can be zero. If $f \neq g$, then the subset

$$L := \{l \in \{0, \ldots, s\} \,|\, I_l \neq J_l\}$$

has at least two elements. Without loss of generality we may assume that $I_\ell = \min\{I_l, J_l : l \in L\}$, i.e., f contains a shortest run — of those letters indexed by L. Then consider the subword g' of g containing all its $\bar{a}$'s, containing at least $I_\ell + 1$ a's in the ℓ-th run of a's, and finally containing, as needed, other copies of a's so that altogether there are at least $\bar{A}+1$ a's. Then, due to the definition, g' is not a subword of f, furthermore, by the number of a's, it is also clear that $\widetilde{g'}$ is also not a subword of f. We know that

$$|g'| \le \max\left\{\left(\left\lceil\frac{A}{2}\right\rceil - 1\right) + 1 + \bar{A},\ 2\bar{A}+1\right\},$$

since the left argument of the maximum includes, within its parentheses, the largest possible value for I_k. If $|g'| \le 2\bar{A}+1 \le 2m$ holds, then there is a contradiction. Therefore this method shows that $D'(f)$ and $D'(g)$ must be different while $\bar{A}+1 \le m$. Continuing the proof from now on (in this section) we assume that

$$\bar{A} > m-1. \tag{4.2}$$

Hence, in this case

$$A = 3m-1-\bar{A} \le 2m-1. \tag{4.3}$$

Denote by $\bar{f}(a,\ell)$ the subword of f containing all a's and the ℓ-th run of $\bar{a}$'s. By our assumptions these are subwords of g, but, as we have just seen, not subwords of $\tilde{g}$. Therefore both f and g can be written in the following forms:

$$f = a^{r_0}\bar{a}^{s_1}a^{r_1}\bar{a}^{s_2}\cdots\bar{a}^{s_t}a^{r_t} \quad \text{and} \quad g = a^{r_0}\bar{a}^{z_1}a^{r_1}\bar{a}^{z_2}\cdots\bar{a}^{z_t}a^{r_t}, \tag{4.4}$$

where r_0 or r_t can be zero, while $r_1,\ldots,r_{t-1}$ and all s_i and z_i are non-zero.

Now we are going to show that for all i we also have $s_i = z_i$ (which, of course, implies that $f = g$).

Let $F \in \{x,y\}^*$ be an arbitrary word and assume it is written in the form

$$F = x^{r_0}y^{s_1}x^{r_1}y^{s_2}\cdots y^{s_t}x^{r_t}, \tag{4.5}$$

where the runs are not empty (except, possibly, the very first and last). That is $r_0, r_t \ge 0$ and all other superscripts > 0. A subword W of F is *well recognizable for the pair* x, y if one can reconstruct exactly which letter of W comes from which x- or y-runs of F. (Reverse complementation is not taken into consideration here. Generally we will ensure separately that the well recognizable subword's reverse complement is not a subword of the original.) It is clear that if the subword W' of F contains W as a subword, then W' is also well recognizable. The subword F_1 containing one letter from each run is clearly well recognizable. Even better, if r_0 and r_t are both non-zero (or, oppositely, both zero), then the reverse complement of this subword is automatically not a subword of F. But when F has a large number of runs (say each run consists of one letter), then one can find much shorter well recognizable subwords.

Proposition 4.1. *Let $W(F)$ be the subword of F defined as follows*:

(I) *$W(F)$ retains at least one x from each x-run.*

(II) *If r_0 or $r_t > 1$, then $W(F)$ contains one x from the respective run and one y from the neighboring y-run.*

(III) *From all other x-runs with precisely two letters, let $W(F)$ contains both.*

(IV) *From all other x-runs with at least three letters, $W(F)$ contains one x from the run and one y from both adjacent runs.*

(En1) *If between two previously chosen y's there are only two-letter x-runs, then keep one x from each of these runs and take one element from each y-run in-between.*

(En2) *From every run of y's, remove all but one.*

Then the resulting $W(F)$ is a well recognizable subword of F for the pair x, y.

(The two last procedures enhance the previously constructed well recognizable words, that give their different kinds of names.) Proposition 4.1 may be thought of as an algorithm, whose six steps are applied sequentially in a single pass. Thus, its validity is evident. Let us remark that without operation (En1) the subword $W(F)$ would be still a well recognizable subword, but this operation decreases the number of letters by one with each application. Note that $W(F)$ never has more letters than the total number of runs in f and neither is it ever shorter than the number of x-runs. However, this construction is sensible for one-letter runs and in their presence it produces well recognizable words with fewer letters than the total number of runs.

Note also that any well recognizable subword of f in Condition (4.4) is also a well recognizable subword of g.

Assume now that $f \neq g$, that is the series $s_1, \ldots, s_t$ and $z_1, \ldots, z_t$ are different. Then the set

$$L := \{l \in \{1, \ldots, t\} \mid s_l \neq z_l\}$$

has at least two elements, since the total number of $\bar{a}$'s are the same in both of our words. Without loss of generality we may assume that $z_\ell = \min\{s_l, z_l : l \in L\}$. At first take the subword f_1 of f containing all its a's and $z_\ell + 1$ $\bar{a}$'s from the ℓ-th $\bar{a}$-run. This word is clearly a well recognizable one, and, due to $A > \bar{A}$, its reverse complement is not a subword of f or g. Therefore, if $A + z_\ell + 1 \leq 2m$, then $f_1 \in D'(f)$ but $f_1 \notin D'(g)$, a contradiction.

If, however, this is not the case, then $|f_1| = 2m + \alpha$ and

$$\begin{aligned} A &= 2m + \alpha - (z_\ell + 1), \\ \bar{A} &= 3m - 1 - A = m - \alpha + z_\ell, \end{aligned} \tag{4.6}$$

where $\alpha \geq 1$. By the minimality of z_ℓ there is another $\bar{a}$-run in f with at least z_ℓ elements. Therefore there are at most

$$t \leq 2 + \bar{A} - (2z_\ell + 1) = m + 1 - (z_\ell + \alpha) \tag{4.7}$$

$\bar{a}$-runs in the word f, and there is at most one more: that is, at most $m + 2 - (z_\ell + \alpha)$ a-runs in f.

Recall that the subword f_1 is not in $D'(f)$ because it has α extra letters and $z_\ell \geq \alpha \geq 1$ (*viz.* (4.6)).

Assume at first that $r_0, r_t > 0$. Then consider the subword f_2 of the word f containing one letter from each run except the ℓ-th $\bar{a}$-run, which contains $z_\ell + 1$ $\bar{a}$'s. This word is well recognizable, and $\widetilde{f_2}$ is not a subword of f or g because they do not contain

enough $\bar{a}$-runs. Furthermore, f_2 is also clearly not a subword of g, since in the ℓ-th $\bar{a}$-run there are too many letters. Due to (4.7) we know that

$$|f_2| \le 1 + 2t + z_\ell \le 1 + 2\,[m + 1 - (z_\ell + \alpha)] + z_\ell = 2m + 3 - 2\alpha - z_\ell \le 2m,$$

since $z_\ell \ge \alpha \ge 1$. Therefore $f_2 \in D'(f)$ but $f_2 \notin D'(g)$, a contradiction.

If $r_0 = r_t = 0$ then we can repeat the previous reasoning since $\widetilde{f_2}$ is not a subword of f or g because there are not enough a-runs in them. If, say, $r_0 > 0$ and $r_t = 0$, then we cannot rule out that the reverse complement of f_2 is a subword of g. In this case there are precisely t $(\le m + 1 - (z_\ell + \alpha))$ a-runs in f. Construct the subword f_3 of f as follows: it contains one letter from each run except the ℓ-th $\bar{a}$-run, which contains $z_\ell + 1$ $\bar{a}$'s. Then f_3 looks like f_2 but it has one fewer element, due to $r_t = 0$. It is a well recognizable subword of f but not a subword of g. Its length is

$$|f_3| = 2t + z_\ell < |f_2|,$$

therefore also $f_3 \in D'(f)$. In general, this would yield a contradiction, but if $r_{t-\ell} > z_\ell$, then $\widetilde{f_3}$ could be a subword of g. But then let f_4 be constructed from f_3 by adding z_ℓ more a letters to the $(t - z_\ell)$-th a-run. This f_4 is clearly a subword of f but not a subword of g or $\widetilde{g}$. Finally

$$|f_4| = |f_3| + z_\ell \le 2m + 2 - 2\alpha \le 2m.$$

Therefore $f_4 \in D'(f)$ but $\notin D'(g)$, a contradiction. The case $\bar{A} < A$ is proved.

4.2. The Case $\bar{A} = A$

In this case we can prove a slightly stronger version of Theorem 2.1: we can suppose that $|f| \le 3m$. Now $|f| = |g|$ is even, i.e., $m = 2k$ and the two words are of the form

$$f = a^{r_0}\bar{a}^{s_1}a^{r_1}\bar{a}^{s_2}\cdots\bar{a}^{s_t}a^{r_t} \quad \text{and} \quad g = a^{R_0}\bar{a}^{z_1}a^{R_1}\bar{a}^{z_2}\cdots\bar{a}^{z_T}a^{R_T}, \tag{4.8}$$

where $r_0 + \cdots + r_t = s_1 + \cdots + s_t = R_0 + \cdots + R_T = z_1 + \cdots + z_T = A = 3k$ and at least one of r_0, r_t and at least one of R_0, R_T is positive, otherwise we exchange the names of f and $\widetilde{f}$, and similarly for g as well. Now without loss of generality we may assume that $r_0 > 0$. Then in g we have $R_0 > 0$. Otherwise the subword $a\bar{a}^A$ of f does not precede g (since there are not enough $\bar{a}$'s after the first a in g, and not enough a's before the last $\bar{a}$ in $\widetilde{g}$).

If $r_t > 0$ also holds, then consider the subword $f_1 = \bar{a}^A a$. If $3k + 1 \le 4k$ then $f_1 \in D'(f)$ but $\widetilde{f_1}$ is not a subword of g, since there are not enough a's after the first $\bar{a}$ in g. Therefore f_1 itself is a subword of g and we have $R_T > 0$; otherwise, there are not enough $\bar{a}$'s before the last a in g. It also means that f_1 is a well recognizable subword of f and g as well. Therefore $r_t = 0 \Leftrightarrow R_T = 0$. (If, however, $|f| \le 4$, then applying Remark 3.6 completes the proof.)

Assume at first that

$$r_t, R_T > 0. \tag{4.9}$$

Denote by F_i the subword of f derived from f_1 by inserting one a from the i-th a-run. If $A \ge 6$ then $F_i \in D'(f)$. These words together, for all i, describe the length of the

$\bar{a}$-runs in f, and all those runs are the complete union of some consecutive $\bar{a}$-runs in g. Repeating the process with g, yielding G_i's, we have the similar correspondence between the $\bar{a}$-runs of f and g. Therefore the $\bar{a}$-run structures of f and g are identical: $t = T$, and $s_i = z_i$; $i = 1, \ldots, t$. (If $A \leq 5$ then Remark 3.6 finishes the proof.) Therefore our words are of the form

$$f = a^{r_0}\bar{a}^{s_1}a^{r_1}\bar{a}^{s_2}\cdots\bar{a}^{s_t}a^{r_t} \quad \text{and} \quad g = a^{R_0}\bar{a}^{s_1}a^{R_1}\bar{a}^{z_2}\cdots\bar{a}^{s_t}a^{R_t}. \tag{4.10}$$

Assume now that $f \neq g$: that is, the series $r_0, \ldots, r_t$ and $R_0, \ldots, R_t$ are different. Then the set

$$L := \{l \in \{0, \ldots, t\} \mid r_l \neq R_l\}$$

has at least two elements, since the total number of a's is A in both words. Without loss of generality we may assume that $R_\ell = \min\{r_l, R_l \colon l \in L\}$. Consider the f-subword $f_2 = \bar{a}^{s_1+\cdots+s_\ell}a^{R_\ell+1}\bar{a}^{s_{\ell+1}+\cdots+s_t}a$. This is clearly neither a subword of g nor of $\widetilde{g}$. Therefore $A + R_\ell + 2 > 4k$, implying that $R_\ell \geq k-1$. Due to the selection procedure for R_ℓ there is another a-run in f of length at least R_ℓ. Then all the other a-runs in f altogether contain $\leq 3k - (2R_\ell + 1)$ letters; hence the numbers of $\bar{a}$-runs are limited: $t \leq 3k - 2R_\ell$. Let the subword f_3 contain one letter from each different run in f, and contain R_ℓ more letters from the ℓ-th a-run. This word has at most $2(3k - 2R_\ell) + 1 + R_\ell = 6k - 3R_\ell + 1 \leq 3k + 4$ letters (here we used $R_\ell \geq k-1$). Since f_3 is a subword of f but does not precede g and this is a contradiction (unless $k \leq 2$, when $|f| \leq 12$ and Remark 3.6 applies; or $k = 3$ and the length of word f's a-runs are 3, 2, 1, 1, 1, 1 which allows again the use of Remark 3.6), Theorem 2.1 is established for this case.

From now on we assume that (4.9) does not hold: that is we have

$$r_t = R_T = 0. \tag{4.11}$$

(Let us recall that at that point we do not know whether the number of runs in f and g are equal or different.) Let $f(a; i)$ denote the subword of f containing all its a's, furthermore one $\bar{a}$ from the i-th $\bar{a}$-run of f; $i = 1, \ldots, t$.

Claim: Every $f(a; i)$ is a subsequence of g or every $f(a; i)$ is a subsequence of $\widetilde{g}$ or both hold.

Indeed, if every $f(a; i)$ is a subsequence of both words then there is nothing to prove. Therefore assume that there is an index i such that $f(a; i)$ is a subsequence of g but not of $\widetilde{g}$. Then for all indices $l \neq i$ the subword $f(a; l)$ is also a subword of g. Indeed, if there is an index l, such that the subword $f(a; l)$ was a subword of $\widetilde{g}$ but not of g, then consider the analogous subword $f(a; i, l)$ of f, containing altogether $A + 2$ letters (all a's and one letter from the i-th and one from the l-th $\bar{a}$-run). This would not be a subword either of g or $\widetilde{g}$, a contradiction, if $A \geq 6$ (if $A < 6$ then Remark 3.6 applies). The Claim is proved.

Therefore we may assume that all $f(a; i)$ are subwords of g; therefore $t \leq T$, and one can make t groups $g_1^*, \ldots, g_t^*$ of consecutive a-runs in g such that the total length of a-runs within g_j^* is equal to s_j. Repeat the whole process for the subwords $g(a; i)$. It still might be necessary to substitute $\widetilde{f}$ for f, but due to (4.11) this already implies that

$$t = T. \tag{4.12}$$

But from this equation it also follows that each $g(a; i)$ is a subword of f, since they are just the image in g of the subwords $f(a; i)$. Therefore we also have $r_i = R_i$ for all i.

Now repeat the whole process for the analogous subwords $f(\bar{a}; i)$ of f. This yields

$$(s_i = z_i, \text{ for all } i) \quad \text{or} \quad (s_i = R_{t-i}, \text{ for all } i).$$

In the first case the proof is complete. Assume that this is not the case. Then the second relation series holds. But repeating the whole process again for the analogous subwords $g(\bar{a}, i)$ then we get that $z_i = r_{t-i}$, for all i. Since we have $r_i = R_i$ it follows that $s_i = z_i$ for all i, which contradicts our assumption, and Theorem 2.1 is proved.

5. Proof of Theorem 2.2

In this section, for conciseness, we will use the notation $\hat{a}$ for both a and $\bar{a}$ and $\hat{b}$ for both b and $\bar{b}$, when the actual value of $\hat{a}$ or $\hat{b}$ is immaterial. With this notation every word of Γ^* can be considered as a word from $\{\hat{a}, \hat{b}\}^*$. Assume that f and g are words in Γ^* of the same length such that

$$|f| = |g| \le 3m + 1 \quad \text{and} \quad D(f) = D(g) = D. \tag{5.1}$$

Without loss of generality we may also assume, due to Remark 3.4, that at least one of the two words, say g, is not self-reverse complementary. Furthermore let

$$p = \max\{|s| \colon s \in D \cap \hat{a}^*\} \quad \text{and} \quad q = \max\{|s| \colon s \in D \cap \hat{b}^*\}.$$

Without loss of generality we can assume that $q \le p$. Let $f(a)$ denote the subword of f consisting of all $\hat{a}$'s. The notation $f(b)$, $g(a)$, and $g(b)$ are analogous. Then, by definition, $|f(a)| \ge p$ and $|f(b)| \ge q$; hence

$$2q \le p + q \le |f(a)| + |f(b)| = |f| \le 3m + 1,$$

and consequently $q \le \frac{3m+1}{2} < 2m$ if $1 < m$. This implies that $|f(b)| = |g(b)| = q$. It also implies that $|f(a)| = |g(a)|$ holds. We remark that $|f(a)|$ may exceed p. (Note that if q is odd, then the subwords containing all $\hat{b}$'s are different from their reverse complements.)

Due to these properties there exist non-negative integers t, T; $i_0, \ldots, i_t$; $r_1, \ldots, r_t$; $j_0, \ldots, j_T$; and $R_1, \ldots, R_T$ such that

$$f = \hat{a}^{i_0}\hat{b}^{r_1}\hat{a}^{i_1}\cdots\hat{b}^{r_t}\hat{a}^{i_t} \quad \text{and} \quad g = \hat{a}^{j_0}\hat{b}^{R_1}\hat{a}^{j_1}\cdots\hat{b}^{R_T}\hat{a}^{j_T}, \tag{5.2}$$

where t can be equal to T, and i_0, i_t, j_0, j_T can be zero, while all other superscripts are nonnegative integers, and, furthermore, where $i_0 + \cdots + i_t = j_0 + \cdots + j_T = |f(a)|$ and $r_1 + \cdots + r_t = R_1 + \cdots + R_T = |f(b)|$. Since $q \le 2m$, the subwords $f(b)$ and $g(b)$ belong to $S(2m, f) = D$; therefore $f(b) = g(b)$ or $f(b) = \widetilde{g}(b)$, or both. Let us remark that we have our general form (4.5) with letters $\hat{a}$ and $\hat{b}$; therefore Proposition 4.1 applies to these words.

For two words w and u denote by $w \simeq u$ if both of $w \prec u$ and $u \prec w$ hold. The following observation will be useful later.

Proposition 5.1. *Assume that $T = t$, $i_k = j_k$ for $k = 0, \ldots, t$ and $r_l = R_l$ for $l = 1, \ldots, t$, and furthermore $f(a) \simeq g(a)$ and $f(b) \simeq g(b)$. Then $f \simeq g$.*

Proof. Suppose instead that $f \neq g$ and $f \neq \widetilde{g}$. We can obtain f by interleaving the runs of $f(a)$ and $f(b)$. Since $f \neq g$ it is easy to see that we must get g from the runs of $\widetilde{f(a)}$ and $f(b)$. If at least one of $f(a)$ and $f(b)$ is self-reverse complementary, then we get $f = \widetilde{g}$ or $f = g$, a contradiction. Suppose now that $f(a) \neq \widetilde{f(a)}$ and $f(b) \neq \widetilde{f(b)}$. Then due to Theorem 1.1 there exists a subword a_* of length at most $\lceil (|f(a)|+1)/2 \rceil$, such that, say, $a_* \leq f(a)$, but $a_* \quad \widetilde{f(a)}$. We get b_* of length at most $\lceil (|f(b)|+1)/2 \rceil$ similarly. Now let f_* be the word obtained from interleaving a_* and b_*. Clearly $f_* \prec f$ but $f_* \quad g$. Hence if $|f| > 7$, then $|f_*| \leq \lceil (|f(a)|+1)/2 \rceil + \lceil (|f(b)|+1)/2 \rceil = \lceil (f+2)/2 \rceil = \lceil (3m+3)/2 \rceil \leq 2m$, a contradiction. (The cases $|f| \leq 7$ are covered by Remark 3.6.) ∎

Next we are going to show that the conditions of Proposition 5.1 hold.

At first we show that the run structures in $f(b)$ and in at least one of $g(b)$ and $\widetilde{g}(b)$ are identical. Denote by $f(b; \ell)$ the subword consisting of all its $\hat{b}$'s and one letter from the ℓ-th $\hat{a}$-run. Since $|f(b; \ell)| \leq 2m$, $m > 1$, this belongs to $D(f) = D(g)$.

Claim: Every $f(b; \ell)$ is a subsequence of g or a subsequence of $\widetilde{g}$ or both hold.

Indeed, if every $f(b; \ell)$ is a subsequence of both words then there is nothing to prove. Therefore assume that for a particular k the word $f(b; k)$ is a subword of, say, g but not of $\widetilde{g}$. Then for all ℓ the words $f(b; \ell)$ are subwords of g as well. Indeed, if there is a $j \neq k$ such that $f(b; j)$ is a subword of $\widetilde{g}$ but not of g, then the f-subword $f(b; k, j)$, defined analogously, is not a subword of either g or $\widetilde{g}$. Because $|f(b; k, j)| \leq (3m+1)/2 + 2$, this yields a contradiction for $m \leq 5$. (The cases $m \leq 4$ are covered by Remark 3.6.) The Claim is proved.

So we can assume that every $f(b; \ell)$ is a subsequence of, say, g. Therefore $t \leq T$, and one can construct t groups $g_1^*, \ldots, g_t^*$ of consecutive $\hat{b}$-runs in g such that the total length of the $\hat{b}$-runs within g_j^* is equal to r_j. Repeat the whole process for the subwords $g(a; i)$. It is possible that we had to substitute $\widetilde{f}$ for f, but this already implies that $t = T$. But from this equation it also follows that each $g(a; \ell)$ can be chosen to be a subword of f since, as we know, the subwords $f(a; i)$ can be found in g. Therefore we also have $r_i = R_i$ for all i and

$$f = \hat{a}^{i_0} \hat{b}^{r_1} \hat{a}^{i_1} \cdots \hat{b}^{r_t} \hat{a}^{i_t} \quad \text{and} \quad g = \hat{a}^{j_0} \hat{b}^{r_1} \hat{a}^{j_1} \cdots \hat{b}^{r_t} \hat{a}^{j_t}, \tag{5.3}$$

where the $\hat{b}$-runs with the same superscripts are identical. Furthermore, we also know that the number of non-empty $\hat{a}$-runs in f and g are equal as well. Indeed, if the multiset $\{i_0, i_r\}$ has no fewer non-zero elements than the multiset $\{j_0, j_r\}$, then the word containing one $\hat{a}$ from the nonempty runs indexed by the first multiset and $f(b)$ establishes this relation. Therefore the number of non-empty $\hat{a}$-runs in f and g is the same, say r': equal to $t-1$, t or $t+1$.

It remains to prove that $f(a) \simeq g(a)$ and that g can be written in a form such that $i_k = j_k$ for all possible k. (Note that if one must interchange g and $\widetilde{g}$ then we will show that in that case $f(b) = \widetilde{f(b)}$.)

5.1. The Case $q = 1$

Let us start with the special case $q = 1$. Now without loss of generality we may assume that both words are written in the form where $\hat{b} = b$ (otherwise we can take the reverse complement form of the word). Now any subword of f containing the letter b should be contained in g in its original form because changing the subword into its reverse complement would change b into $\bar{b}$. Since $|f(a)| = |g(a)|$, $i_0 + i_1 = j_0 + j_1$.

If the multisets $\{i_0, i_1\}$ and $\{j_0, j_1\}$ were different, then there would exist a unique smallest element within them, say, the i_1: we have $i_0 > j_0, j_1 > i_1$. Take a subword u of g of the form

$$u = b\hat{a}^{i_1+1}.$$

This subword clearly does not precede f (there are not enough $\hat{a}$'s after b in the word f). Since $|u| \le (3m+1)/2 \le 2m$, $m > 1$, therefore $D(f) \ne D(g)$, a contradiction. The ordered pairs (i_0, i_1) and (j_0, j_1) coincide. Denote by f_0 the longest simple subword of f ending with b and by f_1 the longest subword of f starting with b. The definitions of g_0 and g_1 are similar. Now f_0 and g_0 are words of the same length, and all their subwords of length $\le 2m$, ending with b coincide as well. Denote by f_0^* and g_0^* the same words without their b terminuses. Then we know that all subwords of length $\lceil (|f_0^*| + 1)/2 \rceil$ of f_0^* and g_0^* are the same over the alphabet $a, \bar{a}$, in the simple subword relation. Application of Theorem 1.1 gives that $f_0^* = g_0^*$ in the original ordering. Furthermore, the same applies to f_1^* and g_1^*; therefore we have proved that $f = g$.

From now on we assume that $1 < q \le (3m+1)/2$. Therefore $|f(a)| = 3m + 1 - q \le 3m - 1$. Now considering the elements $\hat{a}^k \in D$ and applying Theorem 2.1 we get that

$$f(a) \simeq g(a).$$

The only remaining goal is to prove that the $\hat{a}$-structure of the words are the same, i.e., $i_k = j_k$ for all k.

5.2. The Case $1 < q \le m + 1$

Proposition 5.2. *If $1 < q \le m + 1$ and there are two indices $\ell \in \{0, \dots, t\}$ for which*

$$q + i_\ell > 2m, \tag{5.4}$$

then we have $t = 2$, $q = m + 1$, $i_0 = i_1 = j_0 = j_1 = m$.

Proof. Indeed, if $q \le m$ and if there are two distinct indices $k \ne l$ satisfying (5.4) then

$$q + i_l + q + i_k \ge 2m + 1 + 2m + 1;$$

therefore

$$q + i_l + i_k \ge 4m + 2 - q \ge 3m + 2 > |f|,$$

a contradiction.

If, however, $q = m + 1$ and $i_0 = i_1 = m$, then $j_0 = j_1$ as well. Otherwise we would have, say, $j_0 < i_1 < j_1$. Then a g-subword consisting of one letter from the middle $\hat{b}$-run and $i_1 + 1$ letters from the j_1-run is clearly shorter than $2m$ but does not precede f, a contradiction. Let us remark that in this case Proposition 5.1 is applicable directly, and Theorem 2.2 is proved. ∎

If there is precisely one index ℓ satisfying (5.4), then the corresponding run will be called a *long* run, while the other runs are called *short*. Denote by $f^*(b;k)$ the f-subword consisting of all its $\hat{b}$'s and the complete k-th $\hat{a}$-run. For short runs the length of these words is at most $2m$; therefore these belong to $D(f) = D(g)$. Assume for a moment that $f(b) = g(b) \neq \widetilde{g(b)}$. Then $f^*(b;k)$ is not a subword of $\widetilde{g}$ for any short run, and therefore we can find equality of the lengths of the short runs, i.e., $i_k = j_k$ for short runs. Furthermore, because of Proposition 5.2 (i) there is only one $\hat{a}$-run (the ℓ-th), whose length can not be ascertained from the subwords, but then $|i_\ell| = (3m+1-q) - \sum_{k\neq\ell}|i_k| = (3m+1-q) - \sum_{k\neq\ell}|j_k| = |j_\ell|$, which completes the proof in this case. Therefore from now on we assume that

$$f(b) = g(b) = \widetilde{g(b)}$$

holds as well. (We also know that $q = |f(b)|$ is even, but this is not important.)

Case 1. Assume at first that there is a long run in the word f and this is the ℓ-th one. Then g also has at least one long run. Indeed, let u_1 denote an $(2m-q)$-letter subword of the long run. Then the f-subword $f(b)\cup u_1$ belongs to $D(g)$, and the image of u_1 is contained in a long $\hat{a}$-run of g. However, g cannot contain two long runs, otherwise Proposition 5.2 would apply, a contradiction. Therefore g contains exactly one long run and we may assume that f and g contain their respective long runs at the same index ℓ. Let us assume now that $\ell \neq t-\ell$. Then denote by f^*_ℓ the subword containing everything except the ℓ-th and $(t-\ell)$-th $\hat{a}$-runs. This has at most $2m$ letters, and therefore belongs to $D(f)$: that is, it precedes the analogously defined g-subword g^*_ℓ. Similarly g^*_ℓ precedes f^*_ℓ. Consequently we know that $f^*_\ell \simeq g^*_\ell$. This means that

(a) $f^*_\ell = g^*_\ell$, or
(b) $f^*_\ell = \widetilde{g^*_\ell}$,

or both. But all the three possibilities imply that $i_\ell + i_{t-\ell} = j_\ell + j_{t-\ell}$. If (b) does not hold then there is a $k \neq \ell, t-\ell$ such that $\widetilde{f}(b;k)$ is not a subword of $g(b;t-k)$. But since $i_{t-k} \neq 0$, the subword $f(b;k,t-\ell)$ (consisting of all $\hat{b}$'s and one element of the k-th and one element of the $(t-\ell)$-th $\hat{a}$-runs each) which is not longer than $2m$, is therefore a subword of $g(b;k,t-\ell)$, and vice versa, which shows that Proposition 5.1 is applicable. If, however, (b) holds but (a) does not, then there is a k such that $f(b;k)$ is not a subword of $g(b;k)$. Then let u denote an $2m-q-i_k$ element subword of the long run in f. Let f' be the word consisting of u and $f(b;k)$. This is not a subword of g but also not a subword of $\widetilde{g}(b;t-k,t-\ell)$ unless q is very close to m and $j_{t-\ell}$ is also close to m. But then we have a small run-number r and then there is a well recognizable subword of f with at most $2r+1$ letters and repeating the previous reasoning we get the contradiction.

We now come to the case when $\ell = t-\ell$ and t is odd. But then if f^*_ℓ has at most $2m$ letters, which allows us to show as before that $f^*_\ell \simeq g^*_\ell$, and then we can apply Proposition 5.1 again. If this is not the case then we have $q = m+1$ and $i_\ell = m$. If we have at least four non-empty $\hat{a}$-runs then for all $k \neq \ell$ we have $f(b;k,t-k) \simeq g(b;k,t-k)$, showing that $i_\ell = j_\ell$. Furthermore, it is impossible, as usual, that for k_1, k_2 we have $f(b;k_1,t-k_1) = g(b;k_1,t-k_1)$ while $f(b;k_2,t-k_2) = g(b;k_2,t-k_2)$. (We can use the previous technique again.) So Proposition 5.1 is applicable again.

Case 2. Next suppose that there is no long run. Then all $f(b;k) \in D(f) = D(g)$. Assume that for all k the subword $f(b;k,t-k)$ has length $\leq 2m$. Then for all k we have $f(b;k,t-k) \simeq g(b;k,t-k)$. Moreover, as usual, we can show that if there is a k such that $f(b;k,t-k)$ is equal to $g(b;k,t-k)$ but not to its reverse complement; then for all other $l \neq k$ we also have $f(b;l,t-l) = g(b;l,t-l)$. Indeed, if this is not the case then there is a subword f_1 of $f(b;k,t-k)$ with at most $\lceil (i_k + i_{t-k})/2 \rceil$ letters from its $\hat{a}$-runs showing that $f(b;l,t-l) \neq \widetilde{g}(b;l,t-l)$. Similarly, there is a subword f_2 of $f(b;l,t-l)$ with at most $\lceil (i_l + i_{t-l})/2 \rceil$ letters from its $\hat{a}$-runs showing that $f(b;l,t-l) \neq g(b;l,t-l)$. Putting together these two subwords we get a word from $D(f)$ which does not belong to $D(g)$, a contradiction, except that $q = m+1$ and both $\hat{a}$-run pairs contain exactly $m-1$ letters, where m is odd. But again, we can find a well recognizable word with ten letters, and repeating the whole process we are done.

So what remains is that we have an ℓ such that $q + i_\ell + i_{t-\ell} > 2m$. Then for all other $k \neq \ell, t-\ell$ we have $f(b;k,t-k) \simeq g(b;k,t-k)$. (Otherwise we have four non-empty $\hat{a}$-runs, and finding a well recognizable word with eight letters finishes the proof.) Again we can show that, say, $f(b;k,t-k)$ is equal to $g(b;k,t-k)$. Of course, we get that $i_\ell + i_{t-\ell} = j_\ell + j_{t-\ell}$. Then the multisets $\{i_\ell, i_{t-\ell}\}$ and $\{j_\ell, j_{t-\ell}\}$ are the same. Otherwise there would be a clear maximum, say i_ℓ and then $f(b;i_\ell)$ does not precede g, a contradiction. So we are done except that $i_\ell = j_{t-\ell} \neq j_\ell = i_{t-\ell}$. If for all $k \neq \ell, t-\ell$ we have $f(b;k,t-k) = \widetilde{g}(b;k,t-k)$, then we can apply Proposition 5.1 to obtain $f = \widetilde{g}$, or there is a k which does not satisfy this. As usual, we can construct a subword of f with $\lceil (i_k + i_{t-k})/2 \rceil + \lceil (i_\ell + i_{t-\ell})/2 \rceil$ letters from the respective $\hat{a}$-runs which does not precede g: a contradiction, except that again those four runs contain all the $\hat{a}$'s. Repeating the reasoning, we can construct a well recognizable word of length at most, say, 10. So the case $1 < q \leq m+1$ is solved.

5.3. The Case $q > m+1$

In this case we have $p = |f(a)| \leq 2m-1$. Therefore any subword f_k consisting of $f(a)$ and an arbitrary letter from the k-th $\hat{b}$-run belongs to $D(f)$. If $f(a) \neq \widetilde{f(a)}$ then it also means that for all k the subword f_k is a subword of g, and therefore for all k we have $i_k = j_k$. Proposition 5.1 completes the proof.

So we may assume that $f(a) = \widetilde{f(a)}$. Suppose that there is a k such that f_k is a subword of g but not of $\widetilde{g}$. Assume furthermore that there is an ℓ such that f_ℓ is a subword of $\widetilde{g}$ but not of g. (If this second subword does not exist then we already have that the lengths of the $\hat{a}$-runs in f and g are identical.) Let $f_{k,\ell}$ denote the "union" of the former two subwords, then it is a subword of f but not a subword either of g or of $\widetilde{g}$. If $q > m+2$ then $f_{k,\ell} \in D(f)$ therefore it is a contradiction and we are done. But $q = m+2$ can not be true, otherwise $p = 2m-1$ would hold, and therefore $f(a) \neq \widetilde{f(a)}$, a contradiction. Theorem 2.2 is fully proved. ∎

References

1. A.W.M. Dress and P.L. Erdős, Reconstructing words from subwords in linear time, Ann. Combin. **8** (4) (2004) 457–462.

2. A.G. D'yachkov, P.L. Erdős, A.J. Macula, V.V. Rykov, D.C. Torney, C.-S. Tung, P.A. Vilenkin, and P.S. White, Exordium for DNA Codes, J. Comb. Optim. **7** (4) (2003) 369–379.
3. V.I. Levenshtein, On perfect codes in deletion and insertion metric, Discrete Math. **3** (1) (1991) 3–20; Translation in Discrete Math. Appl. **2** (1992) 241–258.
4. V.I. Levenshtein, Efficient reconstruction of sequences from their subsequences or supersequences, J. Combin. Theory Ser. A **93** (2001) 310–332.
5. V.I. Levenshtein, Efficient reconstruction of sequences, IEEE Trans. Inform. Theory **47** (1) (2001) 2–22.
6. M. Lothaire, Combinatorics on Words, Encyclopedia of Mathematics and its Applications **17**, Addison-Wesley, Reading, Mass., 1983.
7. J. Manuch, Characterization of a word by its subwords, In: Developments in Language Theory, G. Rozenberg, et al. Ed., World Scientific Publ. Co., Singapore, (2000) pp. 210–219.
8. I. Simon, Piecewise testable events, Lecture Notes in Comput. Sci. **33** (1975) 214–222.