Válasz opponensi bírálatra

Opponens: Gyimóthy Tibor, egyetemi tanár, az MTA doktora

MTA értekezés címe:
Precíz modell transzformációk tervezése és analízise a modellvezérelt fejlesztésben

Szeretném megköszönni Gyimóthy Tibor Professzor Úr igen alapos, és részletekre kiterjedő bírálatát. Nagyon megtisztelő számomra, hogy magas színvonalúnak és inspirálónak értékeli az értekezésben bemutatott eredményeket. A bírálatban feltett kérdéseket szeretném külön is megköszönni, mert így kitekintést nyújthatok számos olyan kutatási irányra, amelyeket az értekezés benyújtása után kezdtünk el vizsgálni vagy értünk el benne jelentős előrehaladást.

---

1. kérdés: A 3.2. fejezetben szó esik más modellező nyelvek integrálásáról (XML, EMF, BPEL). Az integráció itt konkrétan milyen jellegű és mélységű? A modellek kezelését fekete dobozként jelenik meg vagy mélyebb, szemantikát is érintő integrációt jelent?

Az XML és EMF formalizmusok esetén elsődlegesen szintaktikus integrációt valósítottunk meg, hogy e népszerű, és gyakorlatban is széleskörűen használt modell-leíró formátumokban tárolt modelleket egységes gráfmodellekként tudjuk kezelni a VIATRA2 modelltranszformációs rendszeren belül. Az integráció fő kihívása mindkét esetben az volt, hogy a modellek beolvasását és első konzisztencia ellenőrzését minden esetben az adott alkalmazásterület terminológiáját specifikáló metamodell (EMF) vagy séma (XML) függvényében kellett elvégeznünk, így e feladat is lényegesen meghaladta az egyszerű XML elemzők bonyolultságát.

A BPEL nyelv integrációja kapcsán számos szemantikai vizsgálatot is elvégeztünk a SENSORIA projekt keretében, amelyek – többek között – modellellenőrzésen alapuló helyességellenőrzést [B3], teljesítőképesség és más nem-funkcionális analízist [B4,K12] vagy szolgáltatások automatikus telepítését [J5] célozták. E szolgáltatás-orientált rendszerekben elvégzett kutatások Gönczy László és Hegedüs Ábel PhD disszertációjának alapját képezik.

---

2. kérdés: A VTCL nyelv kialakításánál felmerült-e más elterjedt nyelv alkalmazásának lehetősége, pl. az OCL kiterjesztése? Milyen szempontok alapján döntöttek / mit érdemes megfontolni egy ilyen döntésnél?

A VTCL modell-lekérdező és kényszerleíró nyelve egy deklaratív, gráfmintákon alapuló formalizmus, amely sokkal inkább rokon a szemantikus web területén RDF formátumú tudásreprezentáció lekérdezésére használt SWRL és SPARQL nyelvekkel, mint az OCL-lel. A VTCL nyelvben használt gráftranszformációs formalizmus „legközelebbi rokona” pedig a nyílt forráskódú Drools keretrendszer szabályleíró nyelve. Sajnos, kutatásom kezdetekor (2004-2005-ben) ezek a lekérdezőnyelvek még jellemzően nem léteztek vagy a kutatás igen korai szakaszában jártak, így e „nyelvrokonsági” besorolásom utólagos.

Az OCL szabványos kényszerleíró nyelve ugyan már rendelkezésre állt 2004-ben, a kapcsolódó eszköztámogatás (OCL parszerek és kiértékelők) viszont ekkor még kizárólag a Dresden OCL parszerre korlátozódott. E keretrendszer 2005-ben megjelenő második verziója a Netbeans fejlesztőrendszerhez kapcsolódott, az Eclipse-et támogató verziók csak jóval később jelentek meg. A

VIATRA2 modelltranszformációs rendszer már a kezdetektől az Eclipse keretrendszerbe került integrálásra, így az OCL nyelv elvetése döntően gyakorlati (architektúra, létező eszköztámogatás) megfontolások alapján történt.

Ugyanakkor kiemelném, hogy a VIATRA2 és az EMF-IncQuery gráfminta alapú lekérdezőnyelve az OCL-hez képest jobban támogatja az újrafelhasználhatóságot és a kompozícionalitást, amely nagyon fontos komplex gyakorlati felhasználások esetén. Továbbá a deklaratív leírásmód miatt könnyebb optimalizálni a lekérdezéseket, míg OCL jellegű leírásoknál az élnavigációk jellegzetesen olyan sorrendben hajtódnak végre, ahogy azt a felhasználó definiálta. Így elméleti megfontolások is alátámasztják a deklaratív gráfmintákra eső választásunkat.

---

3. kérdés: Milyen nagyságrendű teljesítménycsökkenést okoz egy generikus transzformáció alkalmazása egy konkrét megfelelőjéhez képest?

A hagyományos objektum-orientált programozási nyelvekben (pl. C++, Java) elérhető generikus konstrukciók feloldása jellegzetesen egy fordítási idejű (előfeldolgozási) lépésben történik meg, így a hatékonyság csökkenése csak a (megnövekedett) fordítási időre korlátozódik.

A VIATRA2 keretrendszer által támogatott generikus transzformációk ugyanakkor más elven működnek. A VIATRA2 modellterében egységesen kerülnek tárolásra a modellek (példányok) és metamodellek (típusok), így a típusparaméterek valójában típusváltozók, amelyek behelyettesítése egy normális gráfmintaillesztési lépésként végezhető el futási időben. A logikai programozás analógiával élve a modelltértől lekérdezhető az **instanceOf(P,T)** predikátum, ahol mind P (példány), mind T (típus) lehet akár bemeneti, akár kimeneti paraméter.

Amennyiben a T típus bemeneti paraméter, úgy a VIATRA virtuális gépben mindössze egy-két (olcsó) többletutasítást jelent a generikus paraméter feldolgozása futási időben, a teljesítménycsökkenés tehát elhanyagolható.

Jelentősebb teljesítménycsökkenés akkor várható, ha T (a típus) is kimeneti paraméter, és nagyméretű modellezési nyelvek (UML, AUTOSAR) metamodelljéből veszi fel az értékét. Ennek a mértéke attól függ, hogy az **instanceOf(P,T)** él P→T irányban vagy T→P kerül illesztésre a keresési terv alapján: első esetben a példány (P), második esetben a típus (T) paraméter van már illesztve az él feldolgozásakor. P→T esetben a példányelem már ismert, innen a típusa olcsón felderíthető. T→P esetben viszont a T típus ismeretében annak összes modellbeli P példányát fel kell sorolnunk, ami jellegzetesen egy igen költséges keresési művelet. Szerencsére ez a tény a VIATRA2 keresési tervet összeállító heurisztikái számára is ismert, így a szabad P paraméterű **instanceOf(P,T)** élek feldolgozása későbbre kerül a keresési tervben (ameddig ez lehetséges).

A generikus transzformációk okozta teljesítménycsökkenés nagyságrendjét pontos mérésekkel sajnos nem vizsgáltuk eddig. Gyakorlati esetekben jelentős lassulást akkor tapasztaltunk, ha a típusok rögzítése után a gráfminta strukturálisan sértené a metamodellt (típushiba esete). Ilyenkor a VIATRA2 keretrendszer a konkrét, nem-generikus esetben már fordítási időben hibát jelez, a generikus transzformáció esetében viszont csak egy szabály illesztésének meghiúsulásából következtethetünk a hibás viselkedésre (ilyenkor a generikus szabály illesztése meghiúsul, de futási idejű hibával nem jár).

4. kérdés: A 3.7. fejezetben láthatunk egy összefoglaló táblázatot a modellező eszközök haladó szintű tulajdonságairól. Milyen funkciókat kellene még megvalósítani, milyen érdekes problémák várnak megoldásra ezen a területen?

Elmondható, hogy az új nyelvi konstrukciók megalkotása helyett a modelltranszformációk kutatásának fókusza az elmúlt években inkább a hatékony végrehajtási technikák (pl. inkrementális, lusta, részleges stb. kiértékelés) felé tolódott el. A 2007-ben készült táblázatban is közölt *kétirányú (bidirectional) transzformációk* terén ma is igen intenzív kutatások folynak (pl. az ETAPS konferencia önálló Bidirectional Transformations workshopjának keretében). A *generikus transzformációk* támogatása ma sem általánosan elterjedt még (de például az ATL [TCJ10], és MOFLON [ALS08] eszközök már támogatják). A *párhuzamos végrehajtás* technikái is megjelentek többek között az ATL, a VMTS és VIATRA rendszerekben (utóbbiakban Mezei Gergely [IM12] illetve Bergmann Gábor [BRV09] munkája nyomán), de még szintén nem általánosan támogatottak.

[ALS08] Carsten Amelunxen, Elodie Legros, Andy Schürr: Generic and reflective graph transformations for the checking and enforcement of modeling guidelines. VL/HCC 2008: 211-218

[TCJ10] M. Tisi, J. Cabot, and F. Jouault. Improving higher-order transformations support in ATL. In Proceedings of the Third international conference on Theory and practice of model transformations (ICMT'10), Laurence Tratt and Martin Gogolla (Eds.). Springer-Verlag, Berlin, Heidelberg, 215-229.

[IM12] G. Imre, G. Mezei: Parallel Graph Transformations on Multicore Systems. MSEPT 2012: 86-89.

[BRV09] G. Bergmann, I. Ráth, D. Varró: Parallelization of Graph Transformation Based on Incremental Pattern Matching. ECEASST 18 (2009)

5. kérdés: Mi motiválta az ILP rendszerek közül az Aleph alkalmazását?

Számos induktív logikai programozást támogató keretrendszer kutatási célra elérhető (Progol, Golem, INTHELEX, FOIDL, stb.), igaz ezek többsége még az 1990-es évek kutatási eredménye, és döntően a Prolog logikai programozási nyelvét használják gazdanyelvként. Tudományos szempontból az INTHELEX talán jobb választás lett volna, mivel támogatja az inkrementális tanulást, és egyszerre több fogalmat is képes megtanulni.

Az Aleph kiválasztását döntően szoftvermérnöki és kevésbé tudományos megfontolások indokolták. Az Aleph a nyílt forráskódú SWI Prolog keretrendszerbe integráltan állt rendelkezésre, amely hatékonyan integrálható az Eclipse keretrendszerrel. Mivel célként tűztük ki, hogy a mintamodellek a VIATRA2 keretrendszerből érkezzenek, és a megtanult szabályokat is ott tegyük közzé, így egyszerre kellett szem előtt tartanunk az ILP keretrendszer „tudományos" és „integrációs" képességeit. E tekintetben az Aleph keretrendszer jó választásnak bizonyult.

6. kérdés: A problémafelvetés szerint a szakterület szakértői nem jártasak a transzformációk leírásában. A módszer első lépéseként létrehozandó metamodell összekötés után azonban egy iteratív folyamat kezdődik, ahol a következtetett szabályok javításra szorulnak. Amellett, hogy a semmiből sokkal nehezebb elkészíteni a transzformációkat, mint pusztán javítani rajtuk, ebben a fázisban a javításhoz mennyire kell ismerni a transzformáció formalizmusát? Van-e esetleg a publikációk óta konkrét tapasztalat valamilyen szakterület specifikus nyelven?

A modelltranszformációk példák alapján történő megkonstruálása elsődlegesen segédeszközt kíván nyújtani a szakterület szakértőinek, hiszen a szabályok teljesen automatikus származtatása nem minden esetben lehetséges. Az alapelvet, miszerint a transzformációs probléma specifikálható a kritikus esetek példaszerű felsorolásával, önmagában intuitívnak tartom a szakterület szakértők számára is. A kutatócsoportunkban az 1990-es évek végén kidolgozott első modelltranszformációk (amelyek UML modellekből különféle hibatűrési modelleket származtattak) is ezt az elvet követték, igaz a transzformációk specifikációja akkor még informálisan (grafikus ábrák megadásával) történt.

Amennyiben a forrás- és célnyelvi modellek grafikus szintakszissal is rendelkeznek (például UML, SysML, BPMN, Petri hálók, stb.), ezek – megfelelő eszköztámogatással – közvetlenül felhasználhatók lehetnek a transzformációs szabályokban is, így a szakértők egy magas absztrakciós szinten láthatják (és szerkeszthetik) a generált transzformációs szabályokat. Ilyen kutatások folytak többek között a Bécsi Műszaki Egyetemen Gerti Kappel vezetésével [WS+07].

Másik lehetőség, ha a szakértők kézzel hajtanak végre egyszerűbb transzformációkat közvetlenül a modelleken, és a transzformáció folyamata (lépései) kerülnek rögzítésre, a transzformációs szabályok így az elemi (kézi) lépések általánosításából erednek (model transformation by-demonstration). Ezt az elvet követik Yu Sun és Jules White kutatásai [SWG09].

E megközelítések sokkal intuitívabb módszert adnak a szakterület szakértőinek a jelen értekezésben bemutatott technikákhoz képest. Ugyanakkor e módszerek közös problémája, hogy a legtöbb transzformációs szabályhoz külön-külön példa megadása szükséges. Az induktív logikai programozáson alapuló általam kidolgozott megközelítés erőssége éppen az, hogy egyetlen forrás- és célmodell párból több transzformációs szabályt is képes származtatni. Így érdekes lehet e két technika ötvözése a jövőben.

[WS+07] Wimmer, M., Strommer, M., Kargl, H., Kramler, G.: Towards Model Transformation Generation By-Example. In: 40th Hawaiian Int. Conf. on Systems Science (HICSS'07). IEEE Computer Society (2007)

[SWG09] Sun, Y., White, J., Gray, J.: Model Transformation by Demonstration. In: 12th Int. Conf. on Model Driven Engineering Languages and Systems (MoDELS'09). LNCS, vol. 5795, pp. 712{726. Springer (2009)

---

1. megjegyzés. Megjegyzem, hogy a modell-specifikus keresésnél használt optimális kifejezés kicsit zavaró, mivel valójában egy megfelelően jó megoldás bemutatásáról van szó. Ezt a szerző meg is jegyzi, de talán érdemes már az elején kerülni a nem egyértelmű megfogalmazást.

Teljes mértékben egyetértek a Bírálóm megjegyzésével, az értekezésben nem igazolom a keresési tervek optimális voltát – sőt a gyakorlati alkalmazások során nincs is szükség erre.

A keresési terv költsége a hozzá tartozó minimális feszítőfa súlya $w(P) = \sum_{i=1}^{n} \prod_{j=1}^{i} w_j$ ahol w_j a j. él súlya a keresési terv által definiált élsorrend szerint. A minimális (irányított) feszítőfa $w(P)$ súlya azonban csak egy becslést ad a lokális keresésen alapuló mintaillesztés várható lépésszámára, amely becslés lehet jó is és rossz is. A gyakorlati mérések azt mutatják [BKG07], hogy általában nagy a korreláció a keresési terv jósága és a mintaillesztő által végrehajtott lépések száma között (tehát kis súlyú keresési tervnél kevesebb mintaillesztési lépés szükséges).

Ugyanakkor egy kellően jó keresési terv gyors megtalálása gyakorlati esetekben jobb megoldást jelenthet, mint az optimum költségesebb megkeresése. Például a Karlsruhei Egyetem kutatói ugyanakkor sikerrel alkalmaztak egy másik közelítő becslést [62], mely szerint a keresési terv összköltségének domináns faktora a legutolsó szorzat, tovább egyszerűsítve az elvégzendő számítást.

[BKG07] Gernot Veit Batz, Moritz Kroll, Rubino Geiß: A First Experimental Evaluation of Search Plan Driven Graph Pattern Matching. AGTIVE 2007: 471-486

---

2. megjegyzés: Az 5.4-es alfejezet végén a szerző ismerteti mely eszköz milyen célra használ inkrementális megoldást. Bár az elvégzett kísérletek alapján látható a nyereség (a módszer kisebb illesztési halmaz esetén különösen hatékony), számomra nem derül ki egyértelműen, hogy a bemutatott megoldás hogyan viszonyul a többi eszköz eredményeihez.

Az inkrementális mintaillesztés kísérleti kiértékelését célzó (az értekezés beadásakor) legfrissebb méréseket az A.2 függelék, és különös tekintettel az A.5-ös ábra tárgyalja, amelyet sajnos nem sikerült az értekezés tartalmi résznek 100 oldalába beleszerkesztenem.

A megcélzott benchmark forgatókönyv szerint a modellező eszközben (pl. UML, AUTOSAR, AADL, SysML, stb.) a tervező nagyméretű modellekkel dolgozik, melyek felett statikus jólformáltsági kényszerek és tervezési szabályok sérülését kell automatikusan és minél gyorsabban detektálni és jelezni a tervező felé (correct-by-construction elv). Méréseink az első validáció és az újravalidáció idejét rögzítették különböző nagyságú modelleken és különféle komplexitású jólformáltsági kényszerek esetén.

A 106. oldal A.5-ös ábráján is látható, hogy a legkomplexebb kényszer esetében (SSG) az OCL és a natív Java alapú, az iparban elterjedten használt megoldások kevesebb, mint 100 000 modellelemig skálázódnak, míg az általunk kidolgozott inkrementális megközelítés 600 000 modellelemre is azonnali végeredményt ad.

A [K6] publikáció óta a statikus jólformáltsági kényszerek újravalidációjának forgatókönyvét mintegy 15 (különféle technológián alapuló) eszközre megmértük (a mérési környezetet itt ismertetjük: http://viatra.inf.mit.bme.hu/publications/trainbenchmark), köztük számos iparban elterjedten használt megoldást is vizsgálva, pl. Drools, MySQL, OpenVirtuoso, Jena, Sesame, RacerPro, stb. A mintegy 6 millió modellelemig futtatott előzetes mérési eredményeink azt mutatják, hogy az általunk kidolgozott inkrementális megközelítés rendelkezik a legegyenletesebb válaszidővel (amely gyakran ténylegesen a legkisebb válaszidőt jelenti különösen nagy modellek és komplex minták esetén).

Az inkrementális modell-lekérdező eszközök teljesítményének hanyatlása legtöbbször a tranzitív lezárt számítás esetén jön elő (amikor az illeszkedéshalmaz jellegzetesen nagy). Ezt kiküszöbölendő kidolgoztunk egy inkrementális megoldást gráfok tranzitív lezártjának kiszámítására [BR+12].

Az inkrementális lekérdezéseket támogató EMF-IncQuery nyílt forráskódú szoftver keretrendszer nemrégiben hivatalosan is az Eclipse Modeling projekt része lett (http://eclipse.org/incquery/), külső partnerek által végzett kísérleti alkalmazásai folynak autóipari és repülőgépipari tervezőeszközökben.

[BR+12] G. Bergmann, I. Ráth, T. Szabó, P. Torrini, D. Varró: Incremental Pattern Matching for the Efficient Computation of Transitive Closure. ICGT 2012: 386-400

7. kérdés: A bevezetőben (1.4.4. fejezet) több szintaktikai és szemantikai követelmény is szerepel a bemutatott terminálási probléma eldöntése mellett. Történtek-e azirányú vizsgálatok, hogy mely más követelmény ellenőrizhető a fentiekhez hasonlóan Petri hálók segítségével?

A Petri hálók és a gráftranszformáció matematikai formalizmusa igen közel áll egymáshoz, hiszen a gráftranszformáció a Chomsky féle nyelvtanokon túl a Petri hálók általánosításaként is felfogható. Sőt, a Prof. Hartmut Ehrig kutatócsoportjában kidolgozott Algebraic High-Level Nets formalizmus [HPR92] a Petri hálók és a gráftranszformáció közös elméleti általánosítását adják. Kapcsolódó kutatások gyakorta előkerülnek az International Conference on Graph Transformation (ICGT) konferencián, és annak PNGT (Petri Nets and Graph Transformation) workshopján.

Modelltranszformációk *szintaktikus helyességének és teljességének* Petri háló alapú vizsgálatára az első általam ismert megoldást Salamon Gábor dolgozta ki még 2001-ben TDK dolgozatában (konzulensek: Pataricza András professzor és én), ahol a kidolgozott módszer egy szabályrendszer sorrendezésének helyességét célozta. A kutatócsoportunkban további kutatások folytak a Petri hálók (és a folyamatszintézis területén rokon formalizmusok) felhasználására együttes helyességellenőrzés és optimalizálási feladatok megoldására, amelyet a Pannon Egyetemmel együttműködésben végeztünk (a főbb kontribútorok: Pataricza András professzor, Varró-Gyapay Szilvia, Friedler Ferenc professzor, dr. Bertók Botond és Nagy Ádám voltak).

A Petri hálók formalizmusa közvetlenül felhasználható a modelltranszformációk *szemantikai helyességvizsgálatára* is, ahogy az alábbi publikációk [W09a,W09b,LV09] demonstrálják. A módszer korlátait leginkább a modellelemek között fennálló komplex strukturális jellegű tulajdonságok (pl. körmentesség) jelentik. Ez természetes következménye a Petri hálók absztrakt formalizmusának, amelyben az adatokat tokenek reprezentálják, amelyek önálló entitásként áramlanak a hálóban.

Összességében tehát elmondható, hogy a legtöbb egyéb követelmény kapcsán is végeztek Petri háló alapú vizsgálatokat.

[HPR92] Hartmut Ehrig, Julia Padberg, Leila Ribeiro: Algebraic High-Level Nets: Petri Nets Revisited. COMPASS/ADT 1992: 188-206

[W09a] Manuel Wimmer, Gerti Kappel, Johannes Schönböck, Angelika Kusel, Werner Retschitzegger, Wieland Schwinger: TROPIC: a framework for model transformations on Petri nets in color. OOPSLA Companion 2009: 783-784

[W09b] M. Wimmer et. al.: Right or Wrong: Verification of Model Transformations using Colored Petri Nets. in Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modeling (DSM´09), Helsinki Business School, Orlando, October 2009

[LV09] J. de Lara and H. Vangheluwe. Automating the Transformation-Based Analysis of Visual Languages. Formal Aspects of Computing, 21, May 2009.

Végezetül szeretném még egyszer megköszönni Bírálom áldozatos munkáját, a megtisztelő értékelést és a mély szakmai kérdéseket, amelyeknek megválaszolásával kitekintést adhattam a közelmúlt kapcsolódó kutatásaira is.

Budapest, 2013. január 31.

Varró Dániel