Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Modelltranszformációk helyességellenőrzése formák analízisével

TDK DOLGOZAT

Készítette:
Semeráth Oszkár
1. éves MSc hallgató, mérnök informatikus

Konzulens:
dr. Varró Dániel
Egyetemi docens, BME-MIT

2011. október 28.

Kivonat

Modellvezérelt tervezés során egy mérnöki modellből modelltranszformáció segítségével egy matematikailag precíz analízis modellt származtatunk, melyen már a fejlesztés korai szakaszában elkezdődhet a rendszer helyességellenőrzése. A gráftranszformációk paradigmája egy intuitív és egyben precíz formális módszert ad e modelltranszformációk specifikációjára. A modelltranszformációkban lévő hibák viszont érvénytelenné tehetik a matematikai modell precizitásából származó előnyöket, így a modelltranszformációk hibamentességének biztosítását célszerű formális verifikációval garantálni.

Valós informatikai rendszerek gyakran potenciálisan végtelen állapottérrel rendelkeznek, amely a formális verifikáció egyik legkomplexebb esete, hiszen a kimerítő állapottér-bejárás közvetlenül nem lehetséges. Így az ellenőrzés végrehajtásához szükséges valamilyen absztrakció alkalmazása. A forma (shape) analízis módszere egy ilyen absztrakciós technikát ad meg. Ennek során gráf alapú modelljeinket ekvivalencia osztályokba soroljuk, és ezeket címkézett gráfok speciális halmazával, formákkal ábrázoljuk. A gráftranszformációs lépések absztrakt megfelelőjét végigvezetve a formákon biztosan véges állapotteret kapunk, melynek analízisével bizonyíthatjuk a gráftranszformációs rendszer helyességét.

Dolgozatom célja, hogy egy egyesített, jól paraméterezhető, gráfelemek ekvivalencia relációján alapuló forma analízissel megteremtsem a lehetőséget arra, hogy a szakirodalomban leggyakrabban használt módszereket keverve vagy egyszerre lehessen alkalmazni, így növelhető legyen azok hatékonysága. Az eljárás egyik legfontosabb és egyben leggyakrabban használt részfeladatai a formák levezethetőségének és ellentmondásosságának felfedése, így ezek végrehajtása kutatásra érdemes terület.

Dolgozatomban megadok egy ekvivalencia relációkkal parametrizálható általános formázási módszert, amellyel figyelembe vehető a bizonyítandó követelmény specifikus tartalma. Leírok egy automatikus módszert a formákon végrehajtandó transzformációs lépés kiszámítására és a formák konzisztenciaellenőrzésére, amely gráfmintaillesztési technikák, ontológiai lekérdezések és dedikált tételbizonyítók kombinált felhasználásán alapul. Fenti módszerek implementációjára architektúrát és megvalósíthatósági tanulmányt adok, mely korszerű szoftverkomponensek (VIATRA2, Pellet és Prover9) felhasználásán alapul.

Mindezekkel egy olyan bővíthető architektúrát adok meg, amely képes létező forma analíziseket és következtető módszereket integráltan alkalmazni. Így elérhető, hogy a kombinált módszer többfajta követelmény bizonyítására legyen képes.

Abstract

In model-driven development, a mathematically precise analysis model is frequently derived from an engineering model by model transformations in order to carry out the formal verification of the system already in an early phase of design. The paradigm of graph transformations offers an intuitive yet formal technique to specify model transformations. However, model transformation errors can invalidate the results of a formal analysis of the mathematical model, so it is advantageous to guarantee the correctness of model transformations by formal verification.

As complex information systems usually have infinite state space, we need to face one of the most complex cases of formal verification since the exhaustive traversal of every possible state is impossible. To tackle this, it is necessary to apply some kind of abstraction in the verification process as provided, for instance, by shape analysis, which categorizes graph based models into a finite number of groups which are represented by a special set of labeled graphs called shapes. Applying the abstract equivalent of graph transformation rules on shapes an abstract but finite state space is obtained, on which one can reason about the correctness of graph languages.

The objective of my report is to enable the combined application of advanced techniques in the literature of shape analysis by proposing a unified, parametrizable shape analysis technique. Here, the most challenging step is to deduce derived properties from shapes and to find their inconsistencies.

In the current report, I define a general method parametrizable by equivalence relations, which is specific to the property to be verified. I propose an automated process to execute the transformation steps on shapes and check their consistency based on combined use of graph pattern matching, queries over ontologies, and dedicated theorem provers. In order to implement this process, I give an architecture and carry out a feasibility study using advanced, state-of-the-art software components (such as VIATRA2, Pellet and Prover9).

The main advantage of this framework is to provide an extensible architecture integrating and combining existing shape analysis and deductive techniques, which allows the verification of a wide range of correctness properties.

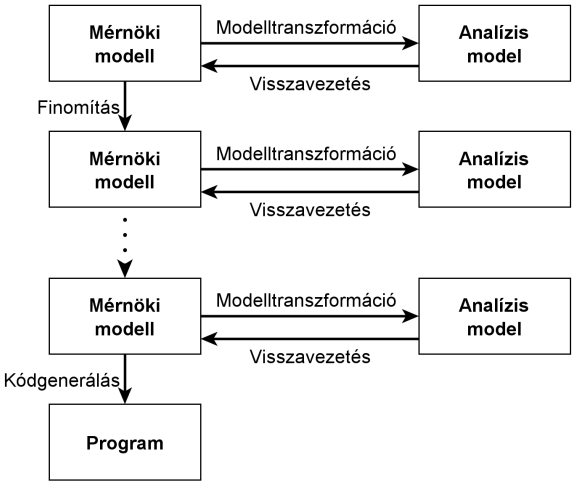
Tartalomjegyzék

1. fejezet

Bevezetés

1.1. Modellvezérelt tervezés

Biztonságkritikus rendszerek tervezése során, melyekben az elkészült programoknak bizonyítottan teljesíteniük kell a tőlük elvárt viselkedést, gyakorta alkalmazzák a modellvezérelt tervezés módszerét [1, 9], melynek menetét a 1.1. ábra mutatja. Ennek során a mérnöki modellből (mint például egy UML aktivitás diagram) modelltranszformációval származtatunk egy matematikailag precíz analízis modellt (például Petri hálót), amin már a tervezés korai szakaszában elkezdődhet a rendszerünk ellenőrzése. Az esetleges koncepcionális hibákat visszavezetjük mérnöki modellünkre, melyet többszörös iterációs lépés mentén finomítunk. A megfelelő felbontású modellt elérve programunkat generáljuk.



1.1. ábra. *A modellvezérelt tervezés menete*

A tervezés során használt modelltranszformációknak olyan feltételeket kell teljesíteniük, melyek által az analízis modell helyességéből következtethetünk a megvalósítás helyességére. Azonban a modelltranszformációkban lévő esetleges hibák érvénytelenné tehetik a matematikai modell precizitásából származó előnyöket, így azok hibamentességének biztosítását célszerű formális verifikációval garantálni.

A gráftranszformációk paradigmája egy intuitív, precíz és hatékonyan végrehajtható formális módszert ad e modelltranszformációk specifikációjára [17], azonban ellenőrzésükkor

komoly elméleti akadályokba ütközünk.

1.2. Feladat bemutatása

1.2.1. Gráfnyelvtanok analízise

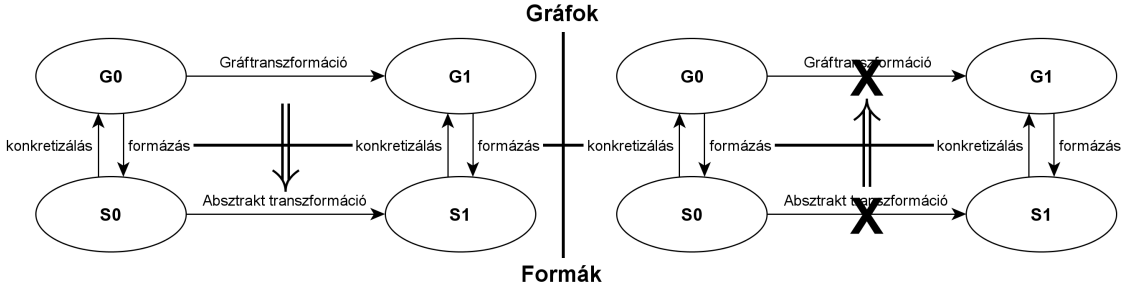
Egy rendszer állapotait modellezhetjük címkézett gráfokkal, a viselkedését gráftranszformációkkal. Egy gráftranszformációs szabályt két gráf ír le: egy bal oldali és egy jobb oldali. A transzformáció egy G gráfon történő végrehajtása során megkeressük G-nek a transzformáció bal oldalára illeszkedő részgráfját, és átírjuk azt a transzformáció jobb oldalára.

A címkehalmazok, egy G_0 kiindulási állapot és gráftranszformációs szabályok R halmaza meghatároz egy gráfnyelvtant. Egy gráfnyelvtan tervezése során felmerülhet az igény annak ellenőrzésére, hogy a kiindulási állapotából a gráftranszformációs szabályokat alkalmazva eljuthatunk-e nemkívánatos tiltott állapotba. A tiltott állapotokat definiálhatjuk úgy, hogy nem teljesül gráfunkra egy P_S biztonsági kritérium (jelölés az angol „safety" szóból származik).

A G_0 kiindulási gráf és az R gráftranszformációs szabályhalmaz meghatároz egy állapotteret, amiben a G_0-ból R-beli transzformációkat alkalmazva elérhető állapotok szerepelnek. Az állapottér elemzése során nagy problémát jelent, hogy mérete potenciálisan végtelen, így az összes elérhető állapot biztonságosságának ellenőrzése kimerítő bejárással lehetetlen.

1.2.2. Absztrakció alapú analízis

Gráfok véges csoportokba történő rendezésével javítható a végtelen állapottér kezelhetetlensége. A csoportokat címkézett gráfok egy speciális halmazával, úgynevezett *formákkal* (angol szakirodalomban *shape*) ábrázoljuk. Egy gráf egy forma által leírt csoportba történő osztályzását *formázásnak* (angolul *shaping*), egy formához tartozó gráfokat a forma *konkretizáltjának* nevezzük. Az 1.2 ábra felső felében gráfok, alsó felében formák szerepelnek, köztük formázások és konkretizálások vezetnek. Fontos megjegyezni, hogy egy formának lehet 0, véges sok, vagy akár végtelen konkretizáltja is.



1.2. ábra. *Konkrét és forma gráfokon értelmezett transzformációk viszonya*

Formákon is értelmezünk transzformációt, amit *absztrakt transzformációnak* nevezünk. Ennek definiálásakor célunk az, hogy felülről becsüljük vele a lehetséges konkrét állapotátmeneteket. Így teljesülnie kell egy feltételnek: ha a G_0-án egy gráftranszformációt alkalmazva G_1-et kapjuk, és G_0-hoz S_0, G_1-hez S_1 forma tartozik, akkor az S_0-án végrehajtva

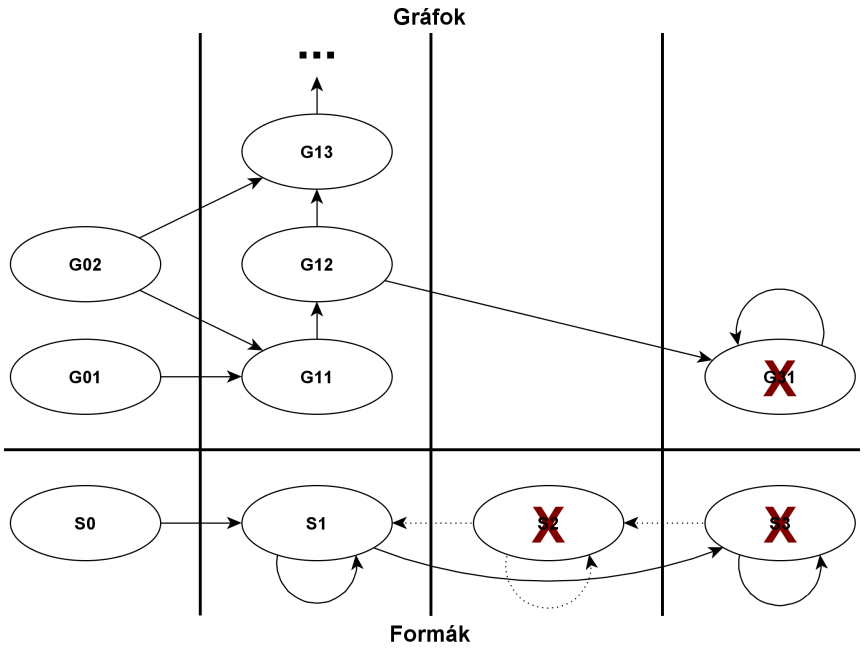
a transzformáció absztrakt megfelelőjét az S_1-et is meg kell hogy kapjuk. Más szavakkal: formák között legalább akkor kell vezetnie formatranszfromációnak, ha léteznek olyan konkretizáltjaik, amelyek között vezet gráftranszformáció. Ezt a megkötést a 1.2 ábra bal oldali diagramja szemlélteti.

A fenti megkötésből következik egy hasznos állítás: ha egy formán nem alkalmazható egy absztrakt transzformáció, akkor a formák konkretizáltjain sem alkalmazható konkrét. Ez az állítás figyelhető meg a 1.2 ábra jobb oldali diagramján.

Az állapotterekhez hasonlóan definiálhatunk absztrakt állapottereket is, melyekben egy kiindulási formából absztrakt transzformációkkal elérhető formák szerepelnek. A formatranszformációk felülről becslő definíciója miatt ebben a gráftranszformációkkal elérhető összes állapotnak megfelelő forma szerepel, illetve az absztrakcióból származó pontatlanság miatt ezeken felül megjelenhetnek más formák is. A konkrét és absztrakt állapotterek viszonyát a 1.3 ábra szemlélteti.

Egy x csoportba absztrahált G_{xy} gráfok egy oszlopba kerültek, a csoportot az S_x forma reprezentálja. A tiltott állapotokat piros kereszt jelzi, a gráftranszformációk során nem létező absztrakciós pontatlanságból eredő átmenetet pedig pontozott vonal.

A formák véges számosságából adódóan lehetőség nyílik az absztrakt terünk felépítésére és elemzésére. Abban az esetben, ha tiltott mintát nem teljesítő állapot nem szerepel benne, bizonyítottá válik, hogy rendszerünk a kiinduló forma egy konkretizáltjából sem kerülhet nemkívánatos állapotba. Ha szerepel, akkor abból általános esetben sajnos semmilyen következtetést nem vonhatunk le.



1.3. ábra. *Konkrét (felső) és absztrakt (alsó) állapotterek viszonya.*

1.2.3. A feladat kihívásai

Az absztrakt állapottereken végzett analízis sikeressége tehát nagyban múlik az alábbi dolgoktól:

1. **Absztrakció pontossága:** Az, hogy egy forma milyen pontosan határozza meg konkretizáltjait, fontos kérdés. Ha túl általános, akkor a pontatlanság miatt lehetetlen lesz megcáfolni a tiltott minta előfordulását, ha viszont szükségtelenül precíz, akkor a formák megnövekvő számossága miatt kivitelezhetetlen lesz az absztrakt állapottér felépítése.

2. **Absztrakt transzformáció pontossága:** A formákon értelmezett transzformációkkal felülről becsüljük a konkrét gráfok között vezető átmeneteket. Ajánlatos viszont arra törekednünk, hogy ezen megkötés mellett transzformációnk minél kevesebb olyan formát adjon eredményül, amely konkretizáltjai között nem létezett volna gráftranszformáció. A felesleges absztrakt állapotátmenetek ugyanis egyrészt megnövelik az állapottér méretét, másrészt indokolatlanul olyan állapotba vihetik azt, amellyel meghiúsulhat a helyességellenőrzés.

3. **Inkonzisztens állapotok szűrése:** Előfordulhat, hogy absztrakt állapotterünk építése során olyan állapotot állítunk elő, amelyről belátható, hogy ellentmondást tartalmaz, s így nem létezik konkretizáltjuk. Az ilyen formák elvezethetik állapotterünket olyan tartományokba is, amelyben alaptalanul jut, így meghiúsíthatja a helyességellenőrzést.

Ezen problémák kezelhetősége érdekében a formázásnak jól paraméterezhetőnek kell lennie. Szerencsés esetben a kiinduló állapot, az elvárt viselkedés során létrejövő állapotok, a gráftranszformációk és a biztonsági kritérium alapján történő paraméterezés esetén olyan formákat kapunk, amelyekben a felhasználás szempontjából érdekes adatokat ábrázolják formáink, az érdektelenek pedig eltűnnek, így csökkentve a formák számosságát.

1.3. Célkitűzések

Dolgozatom célja, hogy egy egyesített, jól paraméterezhető, bővíthető, gráfelemek ekvivalenciarelációján alapuló forma analízissel megteremtsem a lehetőséget arra, hogy a szakirodalomban leggyakrabban használt módszereket keverve vagy egyszerre lehessen alkalmazni, így növelhető legyen azok hatékonysága. Az eljárás egyik legfontosabb és egyben leggyakrabban használt részfeladatai a formák következtetési feladatainak levezetése, és a formák inkonzisztenciájának felfedése, így ezek végrehajthatósága kutatásra érdemes terület.

Dolgozatomban megadok egy ekvivalenciarelációkkal parametrizálható általános formázási módszert, amellyel figyelembe vehető a bizonyítandó követelmény specifikus tartalma. Leírok egy automatikus módszert a formákon végrehajtandó transzformációs lépés kiszámítására és a formák konzisztenciaellenőrzésére, amely gráfmintaillesztési technikák, ontológiai lekérdezések és dedikált tételbizonyítók kombinált felhasználásán alapul. A fenti

módszerek implementációjára architektúrát és megvalósíthatósági tanulmányt adok, mely korszerű szoftverkomponensek (VIATRA2, Pellet és Prover9) felhasználásán alapul.

1.4. A dolgozat felépítése

Dolgozatom a következő felépítést követi:

2. Fejezet: Definiálom a problémakörben alkalmazott fogalmakat, jelöléseket.

3. Fejezet: Bemutatom az általam kidolgozott ekvivalencia alapú absztrakciót, mellyel állapottér-ellenőrzési problémánkat átvezethetjük a formák véges tartományába.

4. Fejezet: Bemutatom, hogy olyan ismert absztrakciós technikák, mint a **TVLA** eszközben megvalósított *predikátumformák* [11] és az irodalomban **neighbourhood shape**ként ismert *szomszédsági formák* [14, 8] megfelelő paraméterezéssel megvalósíthatóak az ekvivalencia alapú absztrakcióban.

5. Fejezet: Megadom az absztrakt állapottér építésének szükséges feltételét, majd részletesen leírok egy ezt a feltételt teljesítő absztrakt transzformátort. A transzformátor az eddig alkalmazott technikákban egyaránt alkalmazott lépésekre épít, ezeket általánosítja, a konkrét lépéseket következtetési szabályok eredményeire bízza.

6. Fejezet: A szükséges számítások elvégzésére bemutatok egy architektúrát, mely több külső következtető rendszerre bízza a feladatokat, és az eredményeiket összesítve használja fel. A fejezet végén megadom, hogy a dolgozat során vizsgált rendszerek mely következtetési feladatok elvégzésére alkalmasak, és melyekre nem.

7. Fejezet: A fejezetben bemutatom, hogyan lehet bizonyos következtetési feladatokat elvégezni ontológiák példányszintű lekérdezésével, hogyan lehet gráfmintaillesztési technikákkal a lehetőségeket hatékonyan szűrni. Megmutatom továbbá hogy néhány feladatot érdemes algebrákkal kiszámoltatni, mint ahogy a szomszédsági formáknál alkalmazott multiplicitásalgebra[12, 8] mutatja.

8. Fejezet: A fejezetben megadok egy leképezést, mellyel formáink elsőrendű logikai állításokká alakíthatóak. Az állításokból automatizált tételbizonyítóval és ellenpélda-keresővel vonok le következtetéseket, melyek elvégzik a szükséges következtetési feladatokat.

9. Fejezet: A fejezetben értékelem a bemutatott absztrakciós technikát. Bemutatom előnyeit és hátrányait az eddig ismert technikákhoz képest, és kitérek pár fejlesztési lehetőségre.

2. fejezet

Elméleti háttér

Ebben a fejezetben bemutatom a gráftranszformációk széles körben alkalmazott és ismert formális módszerét. A fejezet bevezeti a dolgozat során alkalmazott jelölésrendszert és definíciókat, majd bemutatja a paradigma verifikációval kapcsolatos nehézségét.

2.1. Gráfnyelvtanok

2.1.1. Címkézett gráf

A dolgozat olyan véges gráfokat tárgyal, melyneknek csúcsait és éleit adott véges Lab^N és Lab^E halmazok elemeivel címkézzük. A gráfok élei irányítottak, párhuzamos és hurokélek megengedettek.

A *címkézett gráfok* matematikai definíciója, amely tartalmazza a végtelen gráfokra történő kiterjeszthetőséget a következő:

1. Definíció (Címkézett gráf). *Egy* címkézett gráf *az alábbi struktúrával adott:*

$$G = (N, Lab^N, lab^N, E, src, trg, Lab^E, lab^E)$$

ahol:

- N – *A* csúcsok *halmaza.*
- Lab^N – *A* csúcscímkék *halmaza.*
- lab^N – *A* csúcsok címkézésére *szolgáló függvény.* $lab^N : N \mapsto Lab^N$*, azaz minden* $n \in N$ *csúcshoz hozzárendel egy* $lab^N(n) \in Lab^N$ *csúcscímkét.*
- E – *Az* élek *halmaza.* N *és* E *diszjunkt.*
- src – *Az élek* forrása. $src : E \mapsto N$*, azaz minden* $e \in E$ *élhez hozzárendel forrásául egy* $src(e) \in N$ *csúcsot.*
- trg – *Az élek* célja. $trg : E \mapsto N$*, azaz minden* $e \in E$ *élhez hozzárendel céljául egy* $trg(e) \in N$ *csúcsot.*
- Lab^E – *Az* élcímkék *halmaza.*
- lab^E – *Az* élek címkézése. $lab^E : E \mapsto Lab^E$*, azaz minden* $e \in E$ *élhez hozzárendel egy* $lab^E(e) \in Lab^E$ *élcímkét.*

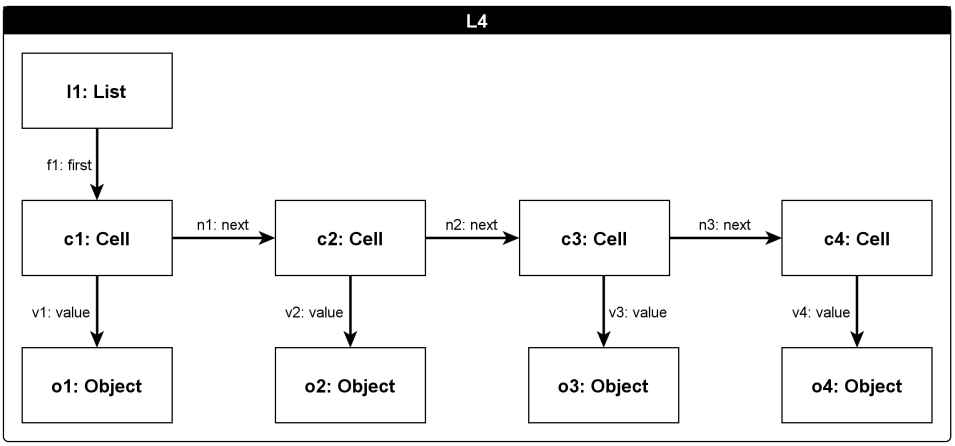
Maga a G *gráf tipikusan véges, ekkor* N*,* Lab^N*,* E *és* Lab^E *véges halmazok. A* Lab^N-

nel és Lab^E-vel címkézett gráfok halmazát $\mathcal{G}(Lab^N, Lab^E)$-vel jelöljük, az összes címkézett gráfot tartalmazó halmazt pedig $\mathcal{G}$-vel. Továbbá jelezzük a összes gráf csúcsait $\mathcal{N}$-nel, éleit $\mathcal{E}$-vel.

Egy gráf struktúrájának egy elemét jelöljük a gráf nevével indexelt elemnévvel. Tehát például N_G a G gráf csúcsainak halmaza.

---

1. Példa A dolgozat során szereplő példákban a 2.1. ábrán látható 4 elemű láncolt listát fogjuk elemezni. A lista minden elemét egy-egy `Cell` cimkéjű gráfcsomópont reprezentálja, a lista láncolását `next` cimkéjű élek jelölik. A benne tárolt `Object` címkéjű értékeket `value` élek kötik a cellákhoz. Egy `List` címkéjű csúcs `first`-tel jelöli meg a lista első elemét. Az ábrán minden csúcsot és élt egyértelmű azonosító jelöl, melyet kettőspont után a gráfelem címkéje követ.



2.1. ábra. *Négyelemű láncolt listát ábrázoló `L4` címkézett gráf.*

A továbbiakban szükség lesz egy olyan leképezésre, amely egy gráf csúcsait és éleit egy másik gráféhoz tudja rendelni. Ennél a függvénynél kikötjük, hogy csúcsokat csúcsokhoz, éleket élekhez rendeljen, és ha egy csúcs egy élnek forrása vagy célja volt, akkor ez igaz legyen a hozzájuk rendelt gráfelemekre is. Ezt *morfizmusnak* nevezzük, pontos definíciója a függelék 11. definíciójában található. Értelmezzünk továbbá a *címkehelyes morfizmust* is, ami annyival szigorúbb fogalom, hogy az összerendelt elemek címkéinek is egyezniük kell.

A morfizmus lehet injektív, szürjektív vagy bijektív függvény. Ezen fogalmak pontos definíciói a függelék 12. definíciójában találhatóak. Két gráf közötti bijektív címkehelyes morfizmus léte meghatároz egy gráfokon értelmezett ekvivalenciarelációt, melyet *izomorfiának* nevezünk.

2. Definíció (Gráf izomorfia). *Két címkézett gráf akkor* izomorf*, ha létezik bijektív címkehelyes morfizmus a két gráf között.*

A későbbiek során két gráfot akkor tekintünk azonosnak, ha izomorfak.

2.1.2. Gráftranszformáció

Egy gráfnak fontos tulajdonsága, hogy megtalálható-e benne egy bizonyos *minta*. Minta alatt egy címkézett gráfot értünk, ami akkor *illeszkedik* egy másik gráfra, ha megtalálható benne részgráfként.

3. Definíció (Gráfmintaillesztés). *Legyenek $P, G \in \mathcal{G}(Lab^N, Lab^E)$. Azt mondjuk, hogy a P minta m mentén illeszkedik G-re, és ezt $P \xrightarrow{m} G$-vel jelöljük, ha m egy P és G közötti címkehelyes injektív morfizmus.*

A *gráftranszformációk*[6] címkézett gráfok feletti műveletek, egy gráfból a transzformációt alkalmazva egy új gráfot kapunk. Egy ilyen műveletet egy gráftranszformációs szabály ír le.

4. Definíció (Gráftranszformációs szabály). Gráftranszformációs szabálynak *nevezzük azt a $Prod = (LHS, RHS)$ párost, ahol LHS és RHS („Left Hand Side" és „Right Hand Side") ugyanazon címkekészlettel címkézett gráfok. P-nek az LHS gráfot a* baloldalának*, az RHS-t* jobboldalának *nevezzük.*

A gráftranszformáció végrehajtásakor megkeresünk egy baloldallal izomorf részgráfot, majd azt kicseréljük a jobboldali gráffal megegyezőre. A dolgozat során az úgynevezett *double-pushout* (DPO)[2, 6] megközelítést követem, miszerint a transzformáció nem hajtható végre, ha ezzel egy él elveszítené forrását vagy célját (azaz nem maradnak „lógó élek"). A művelet precíz megfogalmazását a függelék 13. definíciója tartalmazza.

---

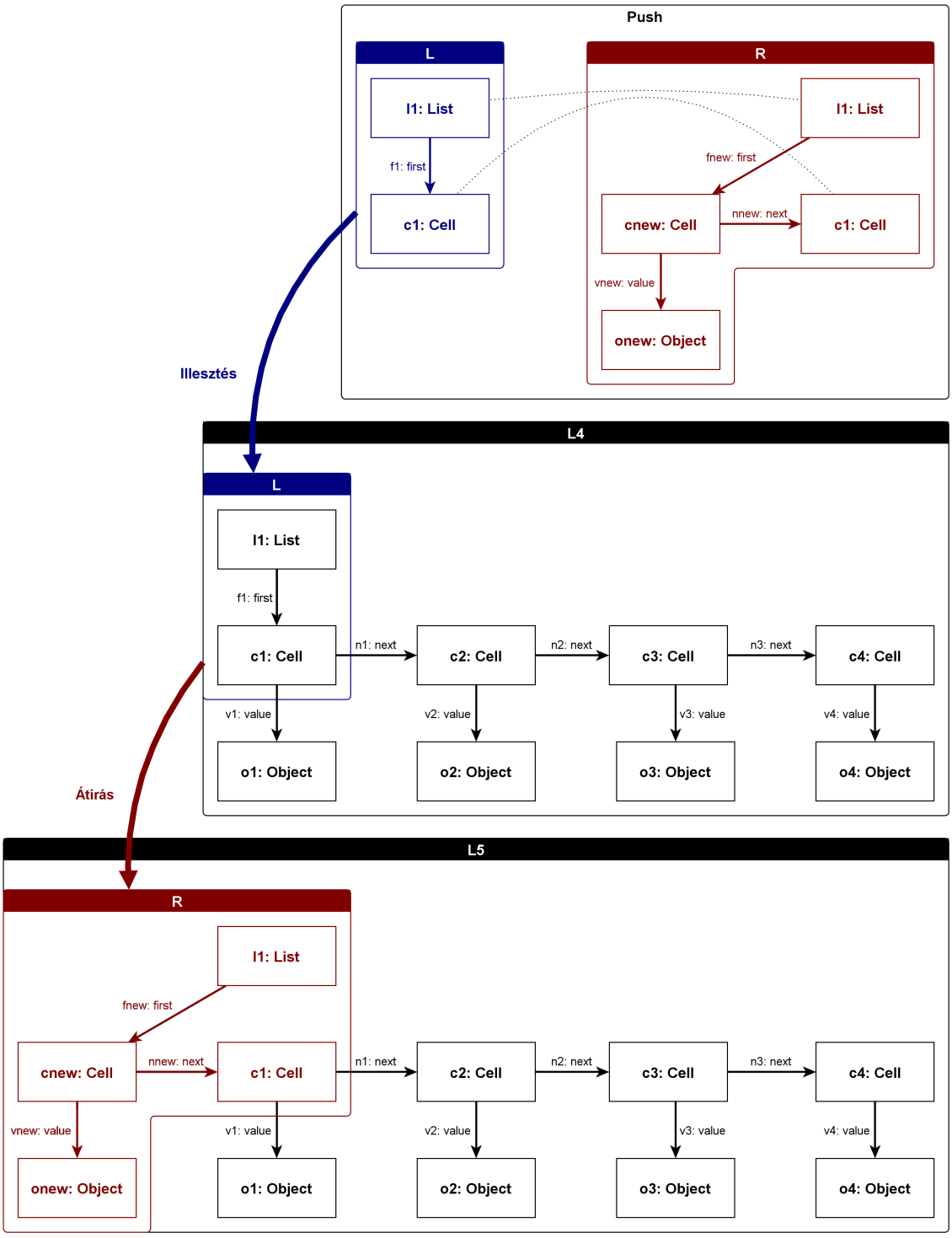
2. Példa A 2.2. ábra tetején látható a `Push` $= (\texttt{L}, \texttt{R})$ gráftranszformációs szabály. Ha ezt a szabályt alkalmazni szeretnénk a négyelemű `L4` gráfon, először keresünk egy m illeszkedést, hogy `L` $\xrightarrow{m}$ `L4` teljesüljön. Ez után megtörténik az átírás, azaz töröljük a törlődő elemeket (mint `f1`), majd felvesszük létrehozzuk az új elemeket (mint `cnew` és `onew` csúcsok és `fnew`, `nnew` és `vnew` élek). A szabályt alkalmazva a 2.2. ábra alján látható 5 elemű listát kapjuk, amelyet jelöljünk `L5`-tel.

2.2. Állapottér ellenőrzése

Egy G_0 kiindulási gráfból R-beli transzformációs szabályokat ismételgetve egy H *elérhető állapotot* kapunk. Elérhető állapotok számításakor megköthetjük, hogy minden köztes állapot teljesítse a $P = P_1, \ldots, P_n$ *jólformáltsági kritériumokat.* Az elérhető állapotot az alábbi módon jelöljük: $G_0 \xrightarrow{R,P} H$. Az elérhető állapot pontos definícióját a függelék 14. definíciója tartalmazza.

A nemkívánatos állapotainkat jellemezzük úgy, hogy nem teljesíti a P_S-sel jelölt biztonsági kritériumot. Ebből következően verifikációnk célja annak eldöntése, hogy létezik-e olyan G_{NS}, hogy $G_0 \xrightarrow{R,P} G_{NS}$ és G_{NS} nem teljesíti a P_S-t. Ennek ellenőrzése céljából kifejtjük az elérhető állapotokat.

Egy kiindulási gráfból elérhető összes gráf *állapotteret* alkot. Az állapotteret egy állapotokkal csúcscímkézett és transzformációkkal élcímkézett gráfként ábrázoljuk, amiben akkor



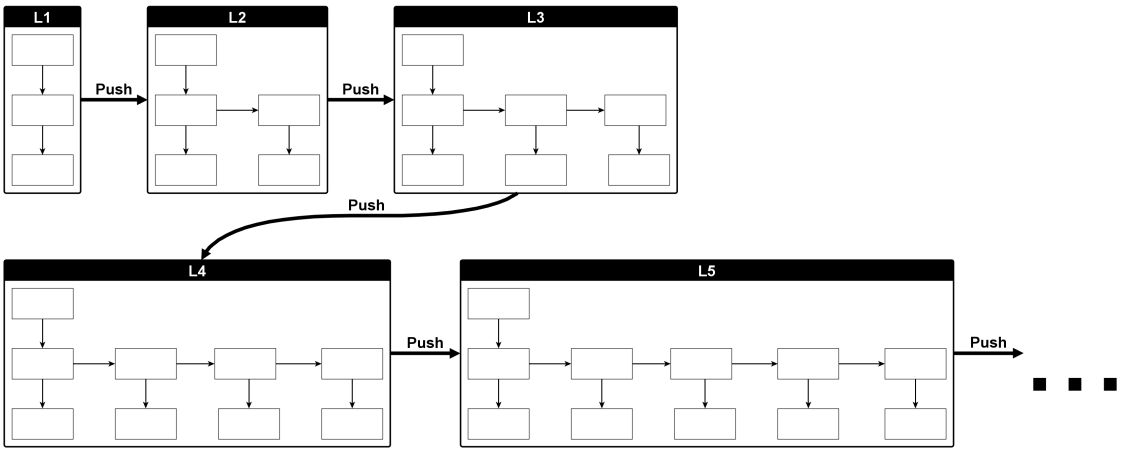
2.2. ábra. *Gráftranszformáció alkalmazása.*

vezet egy P transzformációval címkézett él két csúcs között, ha az egyik csúcs gráf címkéjén alkalmazva a P transzformációt megkapjuk a másik címkéje által ábrázolt gráfot. A precíz definíciót a függelék 15. definíciója tartalmazza.

---

3. Példa Vegyünk kiindulási állapotként a 2.3. ábrán látható `L1` egyelemű láncolt listát, és alkalmazzuk `Push` transzformációt. A transzformáció eredményeként előáll az `L2`, majd ismételt alkalmazásával az `L3` lista, és így tovább. Ennek megfelelően az állapottér az

ábrának megfelelően néz ki.



2.3. ábra. *Az `L1` egyelemű láncolt listából `Push` művelettel generált állapottér.*

Ahogy láthatjuk, az állapottér felépítése után egyértelműen megjelennek az elérhető állapotok, így ez bizonyítaná vagy cáfolná gráfnyelvtanunk helyességét. Azonban a legtöbb valós esetben az állapotterünk végtelen méretű, ezért ennek explicit módon történő felépítése megvalósíthatatlan.

3. fejezet

Formák

A végtelen állapottér okozta kezelhetetlenséget absztrakció alkalmazásával küzdjük le. A fejezetben bemutatok általam kidolgozott, gráfelemek ekvivalenciáján alapuló forma definíciót, amelyben a szakirodalomban használt főbb absztrakciók egységesen ábrázolhatóak.

3.1. Gráfelemek csoportosítása

Gráfok absztrakciójaként gráfjainkat formákhoz csoportosítjuk. Ezt a rendezést vezetjük vissza gráfelemek csoportosítására. Így olyan konstrukciót kapunk, amely ábrázolásában is megőrzi a hozzá tartozó gráfok tulajdonságait.

3.1.1. Kompatibilitási ekvivalencia

A formákkal való ismerkedés első lépésében gráfunk hasonló tulajdonságokkal rendelkező csúcsait véges számú kategóriába csoportosítjuk. Azok a csúcsok kerüljenek egy kategóriába, amelyek egy paraméterként adott csúcsok felett értelmezett véges számú ekvivalenciaosztállyal rendelkező *kompatibilitási ekvivalencia* szerint megegyeznek. Ennek megfelelően formánkban egy tulajdonságcsoport legfeljebb egyszer fog szerepelni, azaz nem lesznek ugyanazzal az ekvivalenciaosztállyal címkézett „hasonló csúcsok", és nem jelennek meg nem szereplő esetek.

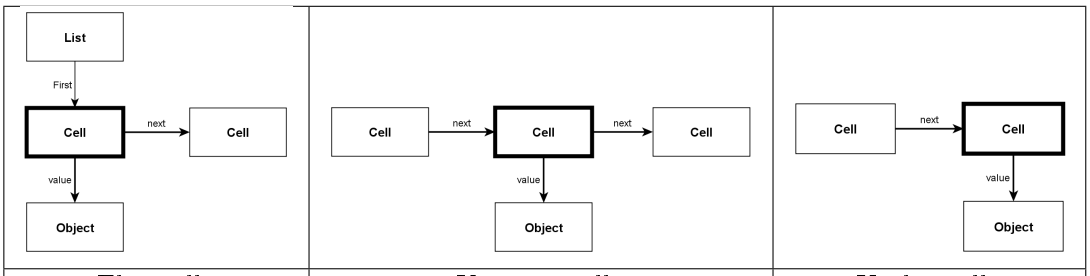
---

4. Példa A gráfcsúcsokon fölött értelmezett $\overset{Neigh}{\sim}$ ekvivalencia akkor teljesüljön, ha 1 sugarú környezetük megegyezik. Négyelemű `L4` listánk `Cell` címkéjű csúcsainak szomszédságát a 3.1. ábra szemlélteti. Megfigyelhető, hogy ezen kategorizálás szerint például különválasztjuk az első `c1` és az utolsó `c4` `Cell`-lel címkézett csúcsot, de a két középső `c2` és `c3` cella összemosódnak, megkülönböztethetetlenek.

A 3.2. ábrán az ekvivalenciának megfelelően egy kategóriába sorolt csúcsokat bekeretezés jelöli.

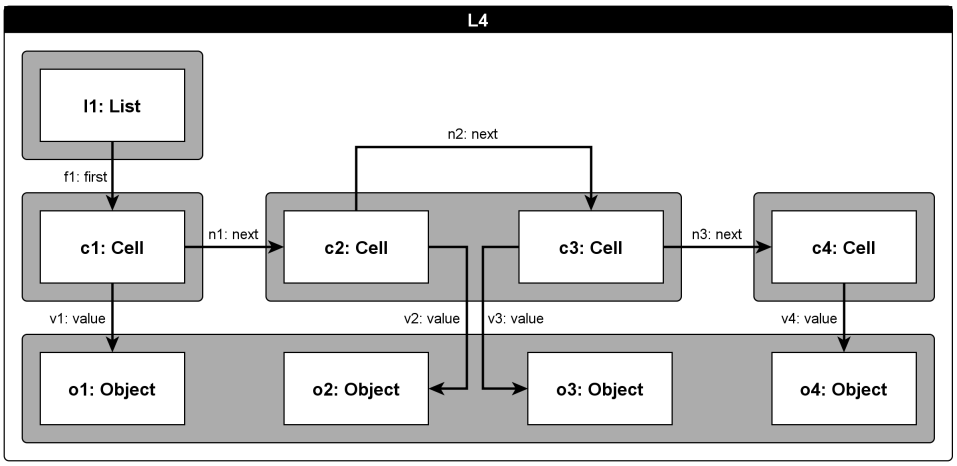
3.1.2. Formák élei

A gráfunk éleit is csoportosítani kell. Az éleknél a csúcsokkal ellentétben nem adunk meg ekvivalenciarelációt, a kompatibilitási ekvivalencia szerint ugyanolyan csúcs-csoportok kö-



Első cella	Középső cella	Utolsó cella
`c1`	`c2` és `c3`	`c4`

3.1. ábra. *A `Cell` címkéjű csúcsok csoportjai. Az első sorban a csúcsok néhány csoportjára jellemző szomszédság látható. Az aktuálisan bemutatott csúcs vastag kerettel van rajzolva. A második sorban a szomszédságnak nevet adunk. A harmadikban a csoportba tartozó csúcsok vannak felsorolva.*



3.2. ábra. *`L4` gráf elemeinek egy csoportosítása.*

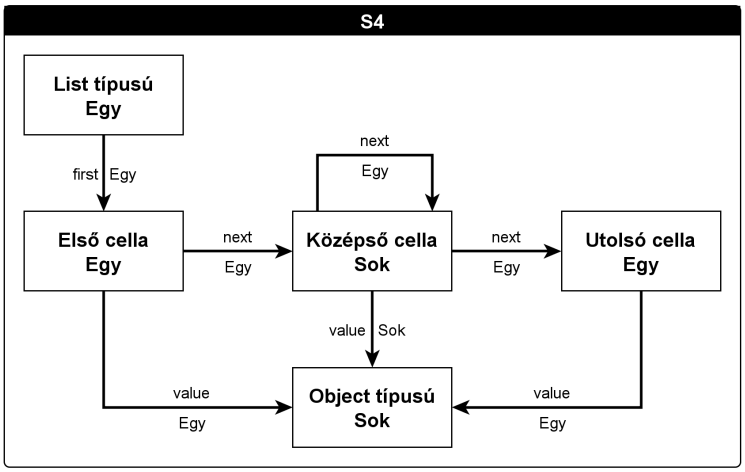
zött futó, azonos címkékkel ellátott éleket rendelünk egy csoportba. Ennek megfelelően formáink csúcsai között nem futhatnak olyan „párhuzamos élek”, melyeknek megegyezik forrásuk, címkéjük és céljuk.

---

5. Példa A 3.2. ábrán egyedül a `v2` és a `v3` élek futnak ugyanazon csúcs-csoportok között, így ezeket összerendeljük. A csúcsok csoportjaihoz hasonlóan itt sem jelennek meg olyan csoportjai az éleknek, amelyeknek nincsenek elemeik, tehát az élekhez is szürjektíven vannak csoportok rendelve.

Ha a 3.2. ábrán látható címkézett gráfnak csak a csúcs- és élcsoportjait jelenítjük meg, a 3.3. ábrához hasonló címkézett gráfot kapunk.

A csúcscsoportok és a köztük futó élcsoportok egy újabb címkézett gráfot alkotnak. Ezt a címkézett gráfot *formának* nevezzük, a gráfelemek formabeli elemekhez rendelését pedig a *formázás morfizmusának*. A formák csúcsait a kompatibilitási ekvivalencia osztályaival címkézzük, úgy, hogy ez a címke megegyezzen a csoportosítás során létrejövő kategóriával. A formák élei az eredeti gráf éleinek címkéivel címkéződnek.



3.3. ábra. *Forma gráf példa.*

3.1.3. Instrumentális ekvivalenciák

Amellett, hogy a konkrét gráfokban előforduló csúcsok és élek tulajdonságait tetszőlegesen pontosan írhatják le ekvivalenciaoszályaik, szükség lehet az egy ekvivalenciaosztályba eső elemek gyűjteményéről, összességéről is állításokat feljegyezni. Például különbséget tehetünk két, egy ekvivalenciaosztályba tartozó csúcshalmaz között, ha a csúcshalmazok számossága eltérő.

A formák ezzel újabb két tulajdonságukkal lesznek paraméterezhetőek. Először formák csúcsai között különbséget szeretnénk tenni egy csúcshalmaz fölött értelmezett ekvivalenciarelációval. Másodszor az élek jelölésére alkossunk egy hármast az élek halmazából, azok forrásainak- és céljainak halmazából, és ezen hármasok fölött is értelmezzünk egy ekvivalenciarelációt. Ezeket az új paramétereket nevezzük *instrumentális* ekvivalenciáknak. A formák a csúcsainak és éleinek címkéi ezzel még egy-egy ekvivalenciaosztályt tartalmaznak.

---

6. Példa A 3.3 ábrán látható forma a csúcshalmazokat két csoportra osztja: egyeleműekre és többeleműekre. A forma csúcsai jelzik, hogy kompatibilitási ekvivalenciaosztályba csak egyetlen (lásd: `Egy` címke), vagy több (`Sok`) csúcs esik a konkét gráfokban. A forma élei is csak azt mutatják, hogy a beleképzett élek számossága egy vagy több-e.

Miután megtudtuk, hogy a középső cella típusú élek között csupán egyetlen next él vezet, a jólformált láncolt listák közül egyedül a 4 elemmel rendelkező felel meg a formának.

3.2. Formák bemutatása

3.2.1. Ekvivalencia relációk

A formák későbbi precíz definiálásához szükség lesz néhány ekvivalenciarelációval kapcsolatos fogalom bevezetésére. Az ekvivalenciarelációkat $\sim$ fogja jelölni, egy a elemmel $\sim$ szerint ekvivalens elemek halmazát $[a]_\sim$, míg K-halmazbeli elemek $\sim$ szerinti ekvivalenciaosztályait $K/\sim$ ábrázolja. Ezen fogalmak definícióját a függelék 16. és 17. pontja tartalmazza.

Reprezentáció

A formák kezelése során szükséges lesz, hogy egy halmaz elemeihez hozzárendeljük a hozzájuk tartozó ekvivalenciaosztályt, illetve ezen osztály szerint csoportosítsuk őket. A formák címkéinek ezeket az ekvivalenciaosztályokat tárolniuk kell, továbbá az osztályok tulajdonságaiból a későbbiekben következtetéseket kell levonnunk. Ezek a feladatok matematikai feldolgozása körülményes, számítási és tárolási költségük magas, ábrázolásuk átláthatatlan.

A problémákra megoldást nyújt a reprezentációk bevezetése, melyeknek feladata, hogy egy-egy ekvivalenciaosztályt egyértelműen jelöljenek. Abban az esetben, ha van olyan függvény, ami egy elemhez hozzárendeli annak reprezentációját, az ekvivalenciák kiértékelését visszavezethetjük a jobb- és baloldal reprezentációjának összehasonlítására. Az ilyen leképezést az ekvivalencia *reprezentáló függvényének* nevezzük, pontos definícióját a függelék 18. definíciója tartalmazza.

Egy H halmazon értelmezett tetszőleges f függvény definiálhat egy olyan $a \sim_f b \Leftrightarrow f(a) = f(b)$ ekvivalenciarelációt, amit f reprezentál. Így a későbbiek folyamán, ha egyszerűbb, akkor reprezentáló függvények fogják definiálni az ekvivalenciákat.

Ekvivalencarelációk kombinálása

Ha van két ekvivalenciánk, és egy halmaz elemeit mindkettő szerint egyszerre hasonlítjuk össze, egy új ekvivalenciát kapunk. Az új relációnk alkalmas arra, hogy finomabb felbontásba osztályozza az elemeket. Az így képzett ekivalenciát a kiindulási E és F ekvivalenciák *szorzatának* nevezzük, és az alábbi módon jelöljük: $\overset{E}{\sim} \times \overset{F}{\sim}$ Az ekvivalenciák szorzatának pontos definícióját a függelék 19. definíciója, a reprezentáló függvényt a 2. tétel mutatja be.

3.2.2. Általános forma definíció

Az ekvivalenciaosztályok kezelhető ábrázolásának bemutatása után eljutottunk a reprezentációkkal paraméterezett forma gráfok matematikai definíciójához:

5. Definíció (Forma). *Jelölje $Comp$ a kompatibilitási reprezentáció, $Instr^N$ az instrumentális csúcs-reprezentációk, Lab^E az eredeti élcímkék és $Instr^E$ instrumentális él-reprezentációk véges halmazát. Legyen továbbá $S \in \mathcal{G}(Comp \times Instr^N, Lab^E \times Instr^E)$. Az S gráf egy $n \in N_S$ csúcsa címkéjének $Comp$-beli részét megkaphatjuk a $comp_S(n)$ függvénnyel, az $Instr^N$-beli részét pedig az $instr^N_S(n)$ függvénnyel. Hasonló módon egy $e \in E_S$ él címkéjének Lab^E-beli része $olab_S(e)$, az $Instr^E$-beli része pedig $instr^E_S(e)$. Összefoglalva:*

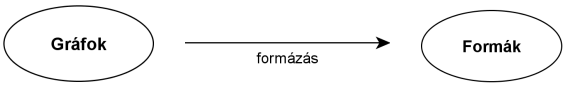
$$\forall n \in N_S \Big(lab^N_S(n) = (comp_S(n), instr^N_S(n)) \Big)$$

$$\forall e \in E_S \Big(lab^E_S(e) = (olab_S(e), instr^E_S(e)) \Big)$$

Az S gráf forma*, ha az alábbi állítások igazak rá:*

- *„Nincsenek hasonló csúcsok", azaz olyanok, amelyeknek ugyanaz a $Comp$-beli címké-*

Paraméter	Értelmezett	Reprezentáció
Kompatibilitási	csúcson	$Comp$
Instrumentális csúcs	csúcshalmazon	$Instr^N$
Instrumentális él	élhalmazon	$Instr^E$



Csúcscímkék: Lab^N
Élcímkék: Lab^E

Csúcsímkék: $Comp \times Instr^N$
Élcímkék: $Lab^E \times Instr^E$

3.4. ábra. *A formázás paraméterei*

jük:

$$\forall n_1, n_2 \in N_S\Big(n_1 \neq n_2 \Rightarrow comp_S(n_1) \neq comp_S(n_2)\Big)$$

- *„Nincsenek párhuzamos élek", azaz olyanok, amelyeknek ugyanaz a forrásuk, céljuk és L_E-beli címkéjük:*

$$\forall e_1, e_2 \in E_S\Big(e_1 \neq e_2 \Rightarrow \\ \neg\Big(olab_S(e_1) = olab_S(e_2) \wedge src_S(e_1) = src_S(e_2) \wedge trg_S(e_1) = trg_S(e_2)\Big)\Big)$$

A $Comp$-pal, $Instr^N$-rel, Lab^E-bal és $Instr^E$-rel címkézett formák halmazát jelöljük a következő módon: $\mathcal{S}(Comp, Instr^N, Lab^E, Instr^E)$. Ha a halmazok egyértelműek, $\mathcal{S}$-t használunk.

Az absztrakt gráfokon történő állapottér felépítésének leglényegesebb feltétele, hogy mérete mindenképpen véges legyen. Ezt a feltételt a formák úgy tudják garantálni, hogy már számosságuk véges. Az erre vonatkozó állítás a függelék 3. tételében olvasható. Így állapottérépítés esetén, még ha az összes lehetséges formát számba kell venni, a művelet befejeződik.

3.2.3. Formázás

A gráfok formákká történő leképezését formázásnak nevezzük. A formázást végző függvény paraméterezésével megadhatóak a kompatibilitási és az instrumentális ekvivalenciák, eredménye pedig a paramétereknek megfelelő címkekészletű forma. A formázásban szereplő paramétereket a 3.4 ábrázolja.

Ez a hozzárendelés rendkívül fontos eleme a gráf absztrakciónak, mivel ez a rész határozza meg az összeköttetést az absztrakt és a konkrét gráfok között. Ebből következően formák tartalmi jelentéséről csak a formázás kontextusának rögzítése esetén van értelme beszélni.

6. Definíció (Formázás). *Legyen G egy Lab^E élcímkékkel címkézett gráf. Legyenek továbbá:*

$\overset{Comp}{\sim} \subseteq \mathcal{N} \times \mathcal{N}$	–	$f_{Comp} : \mathcal{N} \mapsto Comp$ *által reprezentált csúcsok fölött értelmezett kompatibilitási ekvivalencia*
$\overset{Instr^N}{\sim} \subseteq 2^{\mathcal{N}} \times 2^{\mathcal{N}}$	–	$f_{Instr^N} : 2^{\mathcal{N}} \mapsto Instr^N$ *által reprezentált csúcshalmazok fölött értelmezett instrumentális csúcs-ekvivalencia*
$\overset{Instr^E}{\sim} \subseteq (2^{\mathcal{N}} \times 2^{\mathcal{E}} \times 2^{\mathcal{N}})^2$	–	$f_{Instr^E} : 2^{\mathcal{N}} \times 2^{\mathcal{E}} \times 2^{\mathcal{N}} \mapsto Instr^E$ *által reprezentált instrumentális él-ekvivalencia*

Ekkor $shaping(G, \overset{Comp}{\sim}, \overset{Instr^N}{\sim}, \overset{Instr^E}{\sim}) = S$, *ahol* $S \in \mathcal{S}(Comp, Instr^N, Lab_G^E, Instr^E)$, *és* S*-re az alábbi állítások igazak:*

1. *Létezik* G *és* S *között* m *szürjektív morfizmus. Ezt az* m*-et a formázás morfizmusának nevezzük.*
2. $\forall n \in N_G \Big(comp_S(m(n)) = f_{Comp}(n) \Big)$
3. $\forall n \in N_S \Big(instr_S^N(n) = f_{Instr^N}(\operatorname{inv} m^N(n)) \Big)$
4. $\forall e \in E_G \Big(olab_S(m(e)) = lab_G^E(e) \Big)$
5. $\forall e \in E_S \Big(instr_S^E(e) = f_{Instr^E}(\ (\operatorname{inv} m^N(src_S(e)), \operatorname{inv} m^E(e), \operatorname{inv} m^N(trg_S(e)))\ ) \Big)$

Ha az ekvivalenciák, azok reprezentáló függvényei és az élcímkék halmaza adottak, az egyszerűbb $shaping(G)$ *jelölést használjuk.*

Egy adott forma által reprezentált gráfokat a forma konkretizáltjainak nevezzük. A konkretizáltakat megadó függvény a következő:

7. Definíció (Konkretizáció). *Legyenek az alábbiak ekvivalenciarelációk:* $\overset{Comp}{\sim} \subseteq \mathcal{N} \times \mathcal{N}$, $\overset{Instr^N}{\sim} \subseteq 2^{\mathcal{N}} \times 2^{\mathcal{N}}$ *és* $\overset{Instr^E}{\sim} \subseteq 2^{\mathcal{E}} \times 2^{\mathcal{E}}$. *Ekkor egy* S *forma* konkretizációin *az alábbi címkézett gráfokból álló halmazt értjük:*

$$concr(S, \overset{Comp}{\sim}, \overset{Instr^N}{\sim}, \overset{Instr^E}{\sim}) = \operatorname{inv} shape(S, \overset{Comp}{\sim}, \overset{Instr^N}{\sim}, \overset{Instr^E}{\sim})$$

A formázáshoz hasonlóan, ha a többi paraméter egyértelmű, az egyszerűbb $concr(S)$ *jelölést követjük.*

3.3. Összefoglalás

A fejezetben megismerhettük az ekvivalenciaosztályok reprezentációival címkézett formák véges tartományát, melynek absztrakcióját a szakirodalomban megtalálható ismeretanyagokra építve dolgoztam ki. Formázás segítségével problématerünk potencionálisan végtelen számosságú gráfjait leképezhetjük ebbe a tartományba, így elérhetővé válik, hogy helyességellenőrző analízist végezhessünk rajtuk.

Mivel a paraméterként megadott ekvivalenciarelációk szerint csoportosítottuk a gráfelemeket, jól tervezhető, mely elemeket kívánunk összevonni. A következő fejezetből kiderül,

hogy más absztrakció alapú állapottér-ellenőrzést végző eljárásokhoz is alkothatók olyan ekvivalenciák, amelyekkel az általánosított definíciót felparaméterezve megkapjuk az ott leírt absztrakciókat.

4. fejezet

Formák paraméterezése

Az előző fejezetben megadtam a formák és a formázás meghatározását. Ezt követően tekintsünk át pár általánosságban használható ekvivalenciarelációt, melyekkel felparaméterezhetővé válik absztrakciónk.

4.1. Predikátum ekvivalencia

4.1.1. Csúcsokon értelmezett predikátum

Hasznosnak tűnik egy olyan csúcsokon értelmezett predikátummal paraméterezhető ekvivalencia megfogalmazása, amely két osztályba sorolja az elemeket: azokra, amelyekre teljesül a predikátum, és azokra amelyekre nem. A paraméterként megadható predikátumok adott elemi csúcsok és élek címkéjét jelölő unáris és bináris relációkból, éleken értelmezett tranzitív lezártból ($^+$), megszokott logikai összekötőkből ($\neg$, $\wedge$, $\vee$, $\Rightarrow$), logikai egyenlőségvizsgálatból ($=$) és kvantorokból ($\exists$, $\forall$) álló elsőrendű formulák közül kerülnek ki.

8. Definíció (Csúcspredikátum). *Egy $P(n)$ formula csúcspredikátum, ha P-ben egyetlen behelyettesítetlen változó n egy csúcs ($P(n) \Rightarrow n \in \mathcal{N}$).*

---

7. Példa Lássunk csúcspredikátumokra néhány példát:

$isList(n)$	$\Leftrightarrow$	$List(n)$, tehát akkor igaz, ha a csúcs címkéje `List`.
$firstCell(n)$	$\Leftrightarrow$	$Cell(n) \wedge \exists u First(u,n)$, ami akkor teljesül, ha n a sorban az utolsó cella.
$acyclic(n)$	$\Leftrightarrow$	$Cell(n) \wedge \neg Next^+(n,n)$, azaz a csúcs `Cell` típusú és nem része `next` címkéjű élekből álló irányított körnek.
$nonshared(n)$	$\Leftrightarrow$	$Object(n) \wedge \neg(\exists u, \exists v Value(u,n) \wedge Value(v,n) \wedge \neg u = v)$, azaz olyan `Object` típusú csúcs, amelybe nem vezet két különböző csúcsból `value` él.
$reachable(n)$	$\Leftrightarrow$	$\exists u(List(u) \wedge First(u,n)) \vee (\exists v \vee First(u,v) \wedge Next^+(v,n))$, azaz egy `List` címkéjű elem által elsőnek jelölt, vagy az abból `next` élekkel elérhető csúcs.

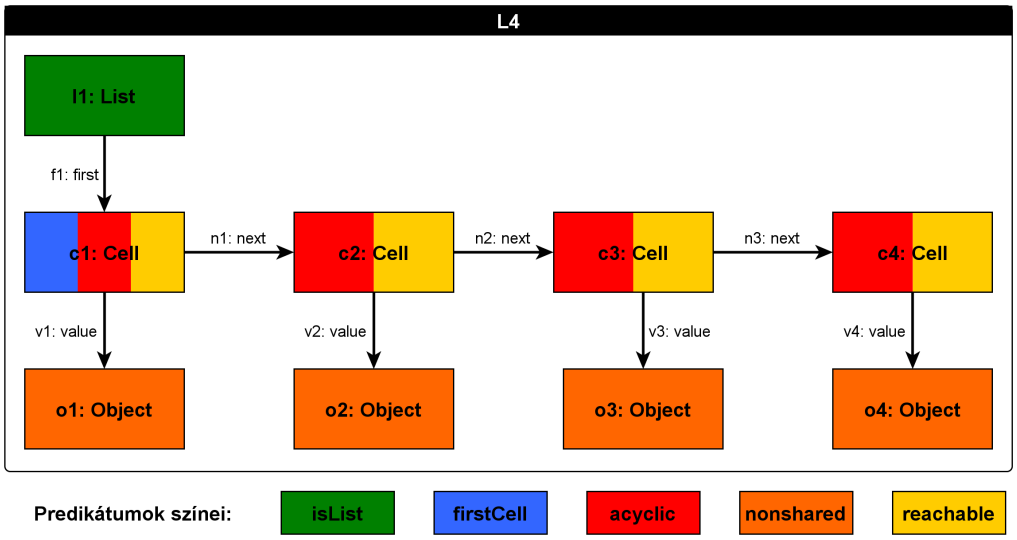
$P(n)$ csúcspredikátum értéke 1, ha az állítás igaz az n csúcsra, és 0, ha nem. Ezek alapján csoportosíthatjuk gráfunk csúcsait:

9. Definíció (Predikátum ekvivalencia). *Legyen P egy csúcspredikátum. Ekkor $\overset{Pred}{\underset{P}{\sim}} \subseteq \mathcal{N} \times \mathcal{N}$ egy olyan ekvivalencia, amelyet a P predikátum reprezentál.*

---

8. Példa Figyeljük meg az 4.1 ábrán látható példát! Az ábrán színezéssel lettek megjelölve gráfunk csúcsai, ha egy predikátum teljesül az adott csúcsra akkor a szín szerepel rajta.

Az *isList* predikátum csak az egyetlen `List` címkéjű `l1` csúcsra teljesül. Láncolt listánk szabályos, azaz minden cellánk teljesíti hogy elérhető (azaz *reachable*) és nem alkot kört (azaz minen cella *acyclic*). Az egyetlen `first` címkéjű `f1` él a `c1` csúcsba vezet, így egyedül ez teljesíti a *firstCell* feltételt. Mivel minden `Object` címkéjű csúcsba egy `value` címkéjű él megy, ezek mindegyike teljesíti a *nonshared* feltételt.



4.1. ábra. *Csúcspredikátumok teljesülését bemutató címkézett gráf.*

4.1.2. Csúcshalmazon értelmezett predikátum

A csúcspredikátumok alkalmazásánál nem elégszünk meg elemek egyedi vizsgálatával. Ebben a szakaszban az lesz a célunk, hogy csúcshalmazokat is tudjunk jellemezni predikátumokkal, így egy új instrumentális ekvivalenciát kapjunk.

Halmazok jellemzésére kevésnek bizonyulnak 0 és 1 értékeink, ezért vezessük be predikátumok igazságtartalmának ismeretlenségét jelölő $1/2$ jelet. Ha megszokott műveleteinket kiterjesztjük bővebb értékkészletünkre (pontos leírás a függelék 1. táblázatában), megkapjuk a *Kleene-féle háromértékű logikát* [10].

Vezessünk be egy új műveletet is: $\cup$. Ha a művelet két operandusa megegyezik, akkor $P \cup Q$ eredménye P (vagy Q is lehetne), ha eltérnek, akkor az eredmény az ismeretlenséget jelző $1/2$.

Ezek alapján a definíció:

10. Definíció (Csúcshalmazon értelmezett predikátum). *Legyen P egy csúcsokon értelmezett predikátum, és $\overset{IPred}{\underset{P}{\sim}} \subseteq 2^{\mathcal{N}} \times 2^{\mathcal{N}}$ ekvivalencia, amit az alábbi $ipred_P : 2^{\mathcal{N}} \mapsto$*

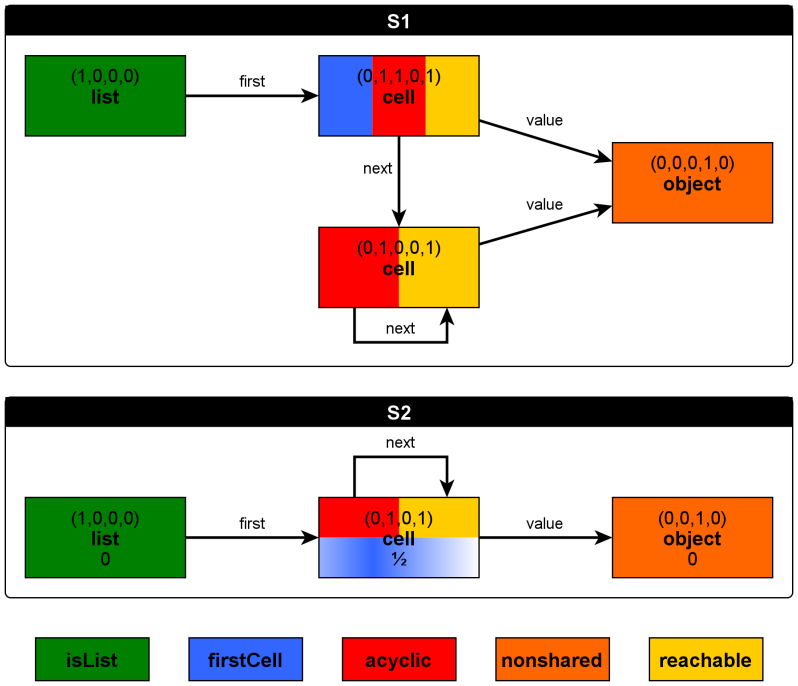
$\{0, 1, 1/2\}$ *függvény reprezentál:*

$$ipred_P(U) = \bigcup_{\forall n \in U} P(n)$$

---

9. Példa Az 4.1. ábrán látható gráfból formákat képzünk, predikátum-ekvivalenciákat alkalmazva kompatibilitásként ($\underset{isList}{\overset{Pred}{\sim}} \times \underset{firstCell}{\overset{Pred}{\sim}} \times \underset{acyclic}{\overset{Pred}{\sim}} \times \underset{nonshared}{\overset{Pred}{\sim}} \times \underset{reachable}{\overset{Pred}{\sim}}$). Így azok a csúcsok kerülnek egy csoportba, amelyekre ugyanazok a predikátumok teljesülnek, azaz ugyanolyan színezéssel rendelkeznek. Így a 4.2 ábrán látható `S1` formát kapjuk.

Mi történik, ha a *firstcell* predikátumot instrumentálisan, és nem kompatibilitásként használjuk? Ekkor `c1` a többi cellától nem különböztetődik meg, és egy formabeli csúcsba képződnek. A $\underset{firstCell}{\overset{IPred}{\sim}}$ ekvivalencia viszont $1/2$-kal jelöli meg a csúcsokat, mivel a predikátum néha teljesül, néha nem. Az így létrejövő `S2` forma a 4.1. ábrán tekinthető meg.



4.2. ábra. *Kompatibilitási és instrumentális ekvivalenciaként alkalmazott firstCell predikátum.*

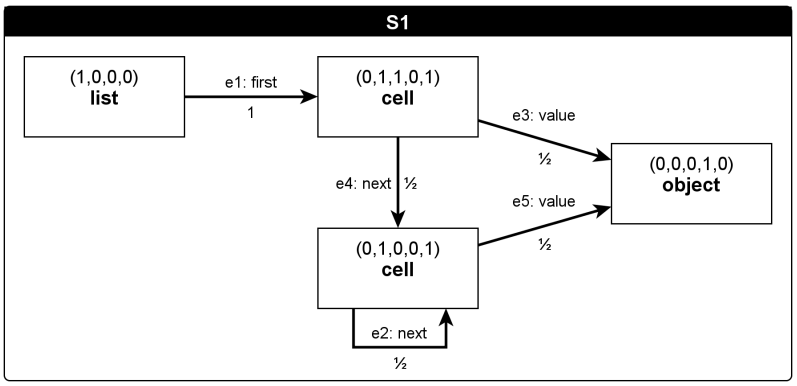
4.2. Bizonytalan élek ekvivalenciája

A formák vizsgálata során hasznos, ha tudjuk, mely esetekben vezet egy él a formabeli csúcsok konkretizáltjai között. Nevezzük azokat a formabeli e éleket *bizonytalannak*, ahol nem teljesül, hogy forrása minden konkretizáltjából vezet él célja minden konkretizáltjába. Ha az e él nem bizonytalan, abból az következik, $src(e)$ és $trg(e)$ kokrét példányai között teljes páros gráf feszül, illetve $src(e) = trg(e)$ esetén olyan teljes gráf, amelynek csúcsain még hurokélek is vannak.

A bizonytalanságuk alapján élek közt különbséget tevő $\overset{Alw}{\sim}$ reláció precíz leírása a függelék 24. definíciójában olvasható

---

10. Példa Ha az `S1` forma csúcsait összegző ekvivalenciával, csúcsait bizonytalansági ekvivalenciával osztályozzuk, akkor a 4.3 ábrán látható formát kapjuk. Az `e1` él instrumentális reprezentációja 1, azaz mindig fut `first` címkéjű él `List` címkéjű csúcs és aközött a cella között, amelyre teljesült a $firstCell$ predikátum.



4.3. ábra. *Összegző csúcsokat és bizonytalan éleket szemléltető forma.*

4.3. Multiplicitás ekvivalencia

4.3.1. Multiplicitás reprezentáció

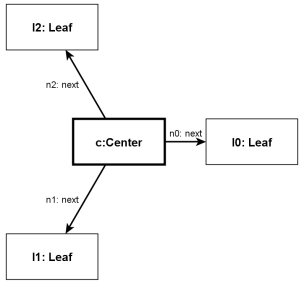
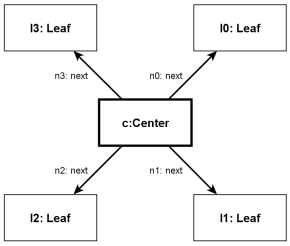
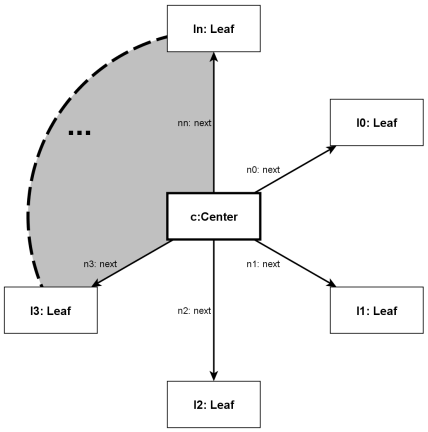
Mennyiségi jellemzők kerekítése céljából bevezetésre kerül a *multiplicitás reprezentáció*. Ennek feladata, hogy a pozitív egész számokat véges számú kategóriákba sorolja. Esetünkben ezek a kategóriák az „egy, kettő, sok" elvén alapulnak, amellyel egy adott maximumig pontosan tároljuk az értéket, azon felül pedig egyszerűen annyit állítunk, hogy sok, és ezt ω-val jelöljük. Ezzel legtöbb esetben a számunkra fontos értékeket pontosan megőrizhetjük.

A multiplicitás, és az egész számokból multiplicitásba képző függvény definíciója a függelék 21. definíciójában látható.

4.3.2. multiplicitás ekvivalencia

Formáinkban érdemes lehet ábrázolni, hogy egy elemcsoportba mennyi elem tartozik. Ezt a számosságot – a véges ekvivalenciaosztály feltételének betartása végett – átalakítjuk multiplicitássá. Ennek következtében két csúcshalmazt akkor tekintünk egyenlőnek, ha számosságuk egyenlő, vagy meghalad egy adott határt. Ezt az m felső korláttal paraméterezhető $\overset{NM}{\underset{m}{\sim}}$ ekvivalenciarelációt *csúcsmultiplicitásnak* nevezzük. Ennek precíz meghatározása a függelék 22. definíciójában olvasható.

Hasonló módon a formák éleinek számossága is leírható egy $\overset{EM}{\underset{m}{\sim}}$ ekvivalenciával (függelék 23. pontja).

3 ágú csillag	4 ágú csillag	n ágú csillag
		

4.1. táblázat. *Csillag alakú gráfok. Egy* `Center` *címkéjű csúcsból „ág" darab* `next` *címkéjű él vezet a* `Leaf` *címkéjű csúcsokba.*

4.4. Szomszédsági ekvivalencia

Ez az ekvivalencia a gráfok csúcsait környezetük alapján hasonlítja össze. Először vezessünk be egy *távolság* paramétert, amivel meghatározzuk a környezetként vizsgált részgráfot: két csúcs akkor legyen izomorf, ha a két csúcstól a paraméterként megadott távolságon belüli részgráfok izomorfak.

Ha megvizsgáljuk a 4.1 táblázatot, láthatjuk, hogy az előbbi definíció véges számú ekvivivalenciaosztály kikötése mellett elégtelennek bizonyul. Az ábrákon látható n-ágú csillaggráfok `Center` címkéjű csúcsai 1 távolság esetén ugyanis n különböző osztályba esnek. Ésszerű lenne, ha egy adott határ feletti ágszámú csillagokat már hasonlónak tekintse ekvivalenciánk. Ezt a határt nevezzük *párhuzamosságnak*.

Multiplicitások segítségével megadható egy olyan definíciót szomszédságra, amely teljesíti, hogy a vizsgált csúcs környezetét r távolságra p párhuzamossággal meg tud különböztetni. Ezt az ekvivalenciát *szomszédságnak* nevezzük és $\overset{Neigh}{\sim}_{r,p}$-val jelöljük. Pontos meghatározását a függelék 25. definíciója adja meg, az őt reprezentáló elemeket és függvényt a 26. és 27. definíció írja le.

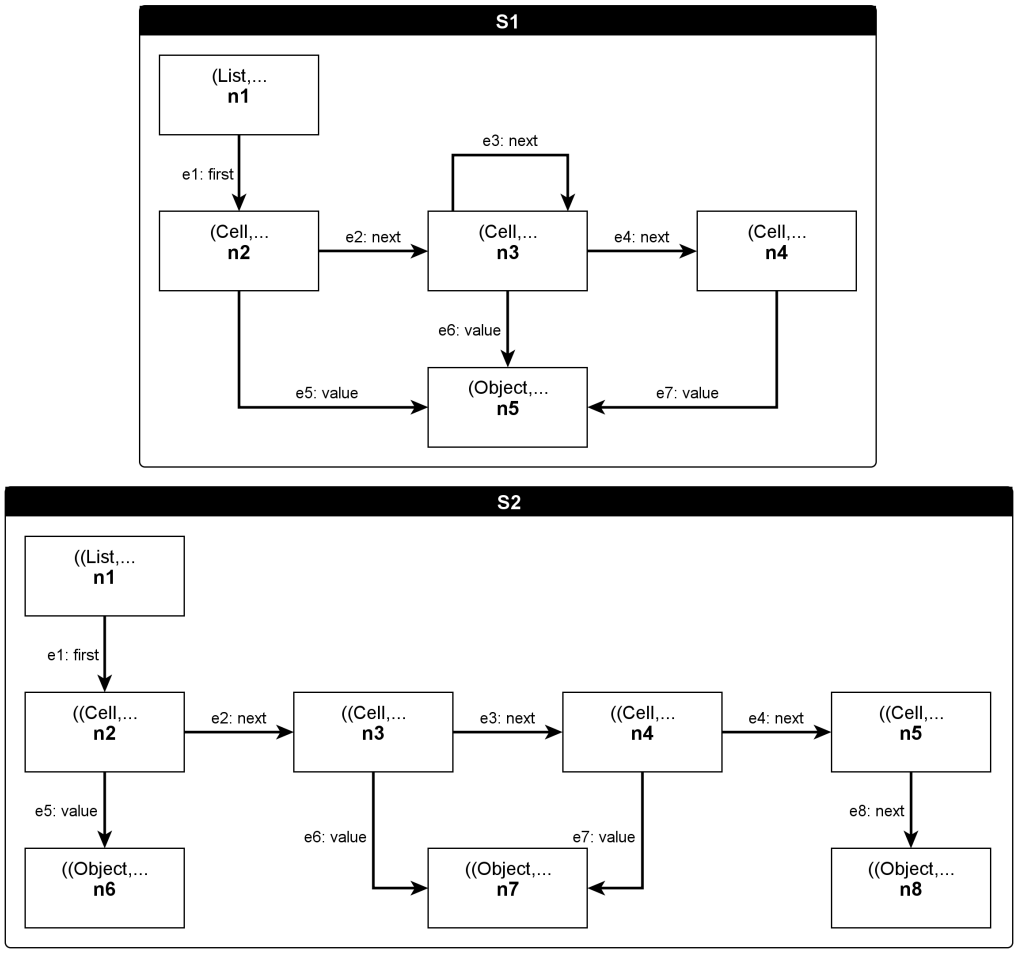
---

11. Példa Az 4.4. ábrán látható két szomszédsági forma. Az `S1` 1, míg `S2` 2 sugarú távolságban különbözteti meg csúcsait, és mindkét esetben egyszeres párhuzamosságot alkalmaztunk.

Látható, hogy `S1`-nél a cellák 3 osztályba sorolódnak: első, középső és utolsó. Ha növeljük a pontosságot (azaz távolabb vizsgáljuk a csúcsok környezetét), kialakulnak olyan osztályok, mint 2. (`n2`) és utolsó előtti (`n3`), valamint első és utolsó objektum (`n6` és `n8`) is.

4.5. Kapcsolódó munkák

A 4.2. táblázatból kiolvasható, mely absztrakciós technikához milyen paraméterezés tartozik. Nem mindegyik absztrakció alkalmazható a címkézett gráfok teljes halmazán, az



4.4. ábra. *Példák szomszédsági formákra.*

ilyen megszorítások a táblázat korlátozásként jelölt oszlopában szerepelnek. Ha ugyanolyan korlátozásokkal szeretnénk végrehajtani a helyességellenőrzést, mint amikkel eredetileg is tették, a korlátozásokat felvehetjük a jólformáltsági kritériumok közé.

A fejezetben bemutatásra kerülnek a predikátum ekvivalenciák, valamint a bizonytalan élek. Ezekkel, valamint 1-es maximális értékű csúcsmultiplicitással megvalósítható a [11] cikkben leírt predikátumformákkal egyenrangú absztrakció. A predikátumformák eredetileg a gráf éleit relációkká képzi, így a párhuzamos élek eltűnnek.

A predikátumformákban kevés (2 vagy 3), de paraméterezés által jól meghatározható csoportra bontjuk a gráfjaink csúcsait. Ennek előnye, hogy kiemelhetjük az analízis szempontjából lényeges, probléma specifikus sajátosságokat. Hátránya, hogy a paraméterezés

Név	**Korlátozás**	**Komp.**	**Instr. csúcs**	**Instr. él**
Predikátumforma	Nincs párhuzamos él	$\prod \underset{P_c}{\overset{Pred}{\sim}}$	$\underset{1}{\overset{NM}{\sim}} \times \prod \underset{P_i}{\overset{Pred}{\sim}}$	$\overset{Alw}{\sim}$
Szomszédsági forma 1	Determinisztikus	$\underset{1,1}{\overset{Neigh}{\sim}}$	$\underset{1}{\overset{NM}{\sim}}$	$\underset{1}{\overset{EM}{\sim}}$
Szomszédsági forma 2	-	$\underset{r,p}{\overset{Neigh}{\sim}}$	$\underset{m}{\overset{NM}{\sim}}$	$\overset{\forall}{\sim}$

4.2. táblázat. *Kapcsolódó munkák*

nem, vagy csak rosszul automatizálható.

Szomszédság alapú kompatibilitási ekvivalencia a [14] (Szomszédsági forma 1) és a [8] (Szomszédsági forma 2) lettek bemutatva. Ezek a absztrakciók multiplicitásokkal teljesednek ki. Az első szomszédsági absztrakció determinisztikus gráfokon van értelmezve. Ez annyit tesz, hogy egy csúcsból nem vezethet ki két olyan él, melyek címkéje megegyező.

A szomszédság alapján történő csoportosítás sajátossága, hogy a csúcsok automatikusan tagolódnak sok szerepkörre. Ennek előnye hogy az analízis során használt ekvivalenciaosztályok könnyen megadhatóak, hátránya, hogy multiplicitások, távolság és párhuzamosság pontosságának növelése általánosan javítják formáink kifejezőerejének pontosságát, így nem törekedhetünk feladatspecifikus megoldásra.

A szomszédsági ekvivalencia másik jellegzetessége, hogy az ekvivalenciaosztályok bonyolult, nagy méretű struktúrák lehetnek. A gazdag ekvivalenciaosztályok miatt szerény paraméterezés mellett is gondot jelenthet a kompatibilitási ekvivalenciák reprezentációinak külön-külön történő tárolása. Érdemes ezért felismerni, hogy szomszédsági reprezentációinknak sok közös részük van. Ha ezeket sikerül úgy ábrázolni, hogy egymás részeire hivatkoznak, a címkék tárolásának erőforrásigénye nagyban lecsökken.

4.6. Összefoglalás

A fejezetben megadtam azokat az ekvivalenciarelációkat, melyekkel a szakirodalomban használt absztrakciók ábrázolhatóak. Ekvivalenciák kombinálásával, szorzatával meglévő ekvivalenciarelációink együttes használata válik lehetségessé. Ezáltal az ekvivalenciaosztályok szaporítása árán elérhető, hogy több szempontból egyszerre osztályozzuk a gráfelemeket, így ötvözhessük módszereink jó tulajdonságait.

5. fejezet

Absztrakt állapottér építése

Ebben a fejezetben meghatározásra kerül az absztrakt transzformáció és transzformátor fogalma. Szerepelni fog a szakirodalmakban előforduló absztrakt transzformátorok egy általánosított leírása, majd megadom a kapcsolódó irodalmakban alkalmazott eljárások lépéseinek megfeleltetését.

5.1. Bevezetés

Az egyszerűbb definíciók végett tisztázzunk előre néhány változót. Legyenek Lab^N és Lab^E véges címkekészletek. $G_0 \in \mathcal{G}(Lab^N, Lab^E)$ kiinduló gráfból R-beli gráftranszformációkat alkalmazva építenénk állapotteret, úgy, hogy eközben minden állapot teljesítsen minden P-beli jólformáltsági kritériumot.

Legyen továbbá:

- $\overset{Comp}{\sim} \subseteq \mathcal{N} \times \mathcal{N}$ – $f_{Comp} : \mathcal{N} \mapsto Comp$ által reprezentált csúcsok fölött értelmezett ekvivalencia
- $\overset{Instr^N}{\sim} \subseteq 2^{\mathcal{N}} \times 2^{\mathcal{N}}$ – $f_{Instr^N} : 2^{\mathcal{N}} \mapsto Instr^N$ által reprezentált csúcshalmazok fölött értelmezett ekvivalencia
- $\overset{Instr^E}{\sim} \subseteq (2^{\mathcal{N}} \times 2^{\mathcal{E}} \times 2^{\mathcal{N}})^2$ – $f_{Instr^E} : 2^{\mathcal{N}} \times 2^{\mathcal{E}} \times 2^{\mathcal{N}} \mapsto Instr^E$ által reprezentált ekvivalencia

Ahelyett, hogy akár végtelenségig építenénk állapotterünket, áttérünk az absztrakt gráfok $\mathcal{S}(Comp, Instr^N, Lab^E, Instr^E)$ halmazára. Ehhez kiszámoljuk a kezdőállapot absztrakt megfelelőjét:

$$S_0 = shaping(G_0, \overset{Comp}{\sim}, \overset{Instr^N}{\sim}, \overset{Instr^E}{\sim})$$

Ezen az S_0 formán fogjuk alkalmazni az R-beli transzformációkat.

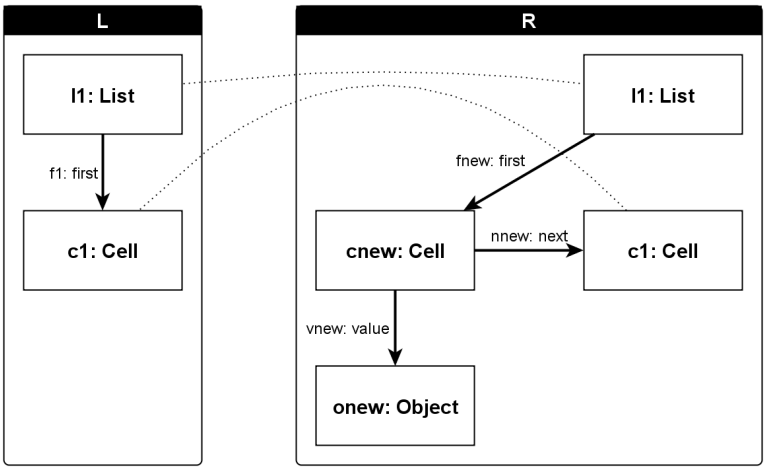
---

12. Példa Az általános definíciókat most is példákkal szemléltetjük. Példáinkban a 4 elemű láncolt listát ábrázoló $G_4 \in \mathcal{G}(\{\texttt{List}, \texttt{Cell}, \texttt{Object}\}, \{\texttt{first}, \texttt{next}, \texttt{value}\})$ kezdőállapotból indulunk ki. Egyedül a `Push` transzformációt használjuk, ami az 5.1 ábrán látható. Példánkban előforduló gráfjainkhoz tartozik még egy jólformáltsági feltétel is, mely megtiltja párhuzamos élek létezését.

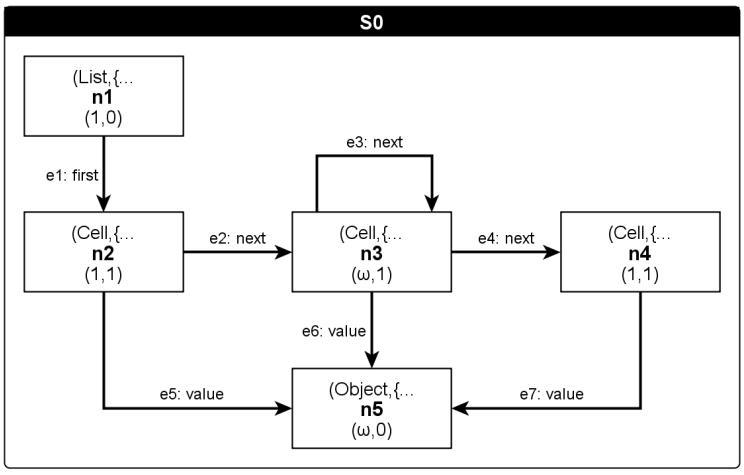
Gráfunk csúcsait szomszédsági ekvivalencia szerint szeretnénk csoportosítani, amelynél 1

a távolság és a párhuzamosság is. A csúcsok instrumentális ekvivalenciájaként egyrészt maximálisan 1 értékű multiplicitást, másrészt elérhetőséget vizsgáló predikátumot választunk. Mivel éleinket a jólformáltsági viszony eléggé korlátozza, úgy gondoljuk, nem szükséges élek instrumentális ekvivalenciájával szaporítani formáink számosságát. Ezt egy olyan $\overset{\forall}{\sim}$ ekvivalencia bevezetésével tesszük meg, ami mindig teljesül, így egyetlen ekvivalenciaosztálya van.

Az előbbiekben megfogalmazott paraméterezéssel alkalmazott formázás során megkapjuk az 5.2 ábrán látható $\texttt{S4} = shaping(G_0, \underset{1,1}{\overset{Neigh}{\sim}}, \underset{1}{\overset{NM}{\sim}} \times \underset{reachable}{\overset{Pred}{\sim}}, \overset{\forall}{\sim})$ formát.



5.1. ábra. *A* $\texttt{Push} = (L, R)$ *transzformációt leíró diagram.*



5.2. ábra. *Az* $\texttt{S0}$ *forma.*

5.2. Absztrakt gráftranszformációk

Absztrakt állapotterünk építése során a célunk, hogy minden konkrét állapotátmenetet rögzítsünk, ezért a formatranszformációknak felül kell becsülniük a gráftranszformációkat. A formatranszfromációk pontos leírása a függelék 28. definíciójában olvasható. Amint lát-

ható, a transzformáció egyértelmű eredménye nincs meghatározva, ezt a *transzformátorok* definiálják. Így az absztrakt állapotterünk kifejtése előtt meg kell adnunk egy, a feltételeknek megfelelő transzformátort.

Általában a kötelező állapotátmenetek mellett szerepelnek olyanok is, amelyek az absztrakcióból származó bizonytalanságból származnak. Mivel a felesleges állapotátmenetek meghiúsíthatják helyességellenőrzésünket, két transzformátor közül azt tekintjük pontosabbnak, amelynek eredménye kisebb halmaz.

A transzformátorok közül van egy, ami legalább olyan pontos, mint a többi, és ezt tökéletes transzformátornak [3] nevezzük (függelék 29. definíciója). A tökéletes transzformátor működésének menete úgy képzelhető el, hogy a formánknak kiszámítjuk az összes konkretizáltját, azokon végrehajtjuk a gráftranszformációkat, melyeknek eredményeit újra formázzuk. A módszer kivitelezhetlensége az összes konkretizált megszerzéséből ered, mivel ezek számossága akár végtelen is lehet.

5.3. Absztrakt transzformátor leírása

Ebben a szakaszban megadásra kerül egy olyan transzformátor váza, amely a használt ekvivalenciák tekintetében hatékonyan működő következtető rendszer mellett használható formatraszformációk végrehajtására. A transzformáció menete a következő lesz:

1. Absztrakt illesztés(5.3.1): Megkeressük, hogy formánk mely csúcsaira és éleire illeszkedhet a transzformáció baloldala.

2. Fókusz(5.3.2): Az illeszkedés mentén konkretizáljuk formánk azon részét, amelyre illeszkedhet a baloldal.

3. Szűrés(5.3.3):A fókuszált gráfunk ellentmondhat a formánk reprezentációiban tárolt ismeretnek. Ezeket az eseteket ki kell szűrni.

4. Transzformációs lépés(5.3.4): Végrehajtjuk a transzformációs lépést a formánk konkrét részén. Ha kell, az absztrakt gráfelemek reprezentációit megváltoztatjuk.

5. Normalizálás(5.3.5): A transzformációs lépés után a transzformáció jobboldalára fókuszált részgráfot kapunk. Ezt vissza kell alakítani formává.

6. Konzisztenciavizsgálat(5.3.6): Ellenőrizzük, lehet-e a létrejövő formának konkretizáltja.

A transzformáció lépéseikor több részeredményt is kaphatunk. A kapott részeredmények mindegyikével tovább kell számolnunk, az utolsó pont végén megmaradt formák lesznek a transzformáció eredményei.

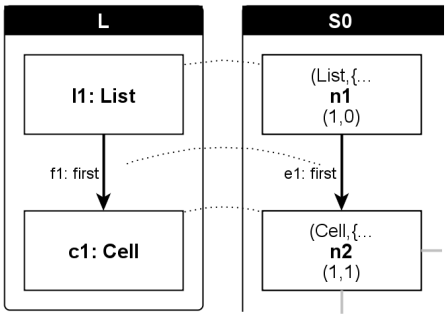
A következő definíciókban bevezetünk egy új szóhasználatot: *A összeegyeztethető B-vel*. Ennek jelentése az, hogy A absztrakt gráfelem reprezentációiból nem lehet, nem tudjuk, vagy nincs elég erőforrásunk belátni, hogy B ne lehetne eleme az A által képviselt csoportnak.

5.3.1. Absztrakt illesztés

Absztrakt illesztés során megpróbálunk egy konkrét gráfot illeszteni formánkra. Az illesztéskor a gráfelemek illeszthetőségét elemenként vizsgáljuk, azaz a gráfunk csúcs és élcímkéit össze kell vetnünk formánk reprezentációival. Olyan morfizmusokat keresünk, amelyeknél az összerendelt élek címkéi megegyeznek, és a csúcscímkék összeegyeztethetőek képük reprezentációjával. Pontos leírás a függelék 30. definíciójában található.

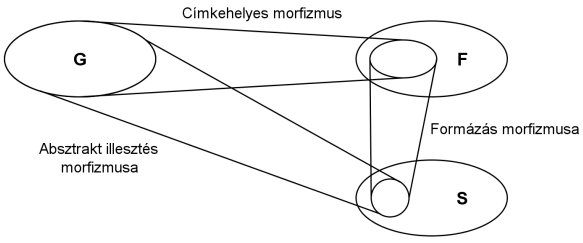
---

13. Példa A 5.3 ábrán láthatunk egy példát absztrakt illesztésre, a `Push` transzformáció baloldalát szeretnénk m_0 morfizmus mentén illeszteni `S0`-ra. Mivel az `S0` formánkban csupán egyetlen `first` címkéjű él van, az ábrán látható illeszkedésen kívül minden más kizárható. A `List` és `Cell` címkéket `n1` és `n2` reprezentációi nem cáfolják meg, így m_0 absztrakt morfizmus.



5.3. ábra. *A `Push` transzformáció baloldalának illesztése az `S4` formára. Az m_0 absztrakt illesztést pontozott vonal jelöli.*

Az absztrakt illesztés legfontosabb tulajdonsága, hogy lefed minden konkretizáltakban előforduló morfizmust. Ha létezik G és F gráfunk között egy m morfizmus, és $S = shaping(S)$ esetén m_s a formázás morfizmusa, akkor $m_a = m_s \circ m$ egy G és S közötti absztrakt morfizmus. Az összefüggést a 5.4 ábra szemlélteti.



5.4. ábra. *Címkehelyes morfizmus, formázás morfizmusa és absztrakt morfizmus viszonya.*

Az absztrakt illesztés keresése felvet egy következtetési feladatot:

Konkrét illesztés: Egy következtető rendszernek meg kell adnia egy olyan (lehetőleg minimális) illeszkedéshalmazt, amelyben az összes absztrakt illesztés szerepel.

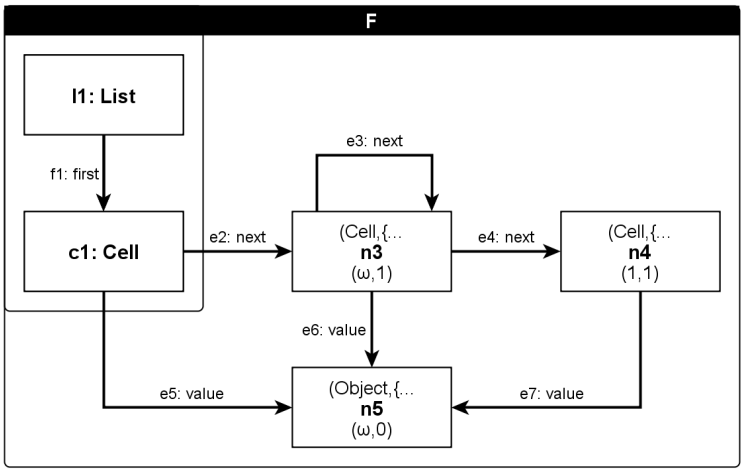
5.3.2. Fókusz

A tökéletes absztrakt transzformátor az összes konkretizálton végrehajtaná a transzformációt, ám ez megvalósíthatósági korlátokba ütközik. A probléma áthidalására kompromisszumot kell kötnünk, és gráfunknak csak azon részét konkretizáljuk, amelyen a transzformációt végrehajtanánk. Ezt a műveletet *fókuszálásnak* nevezzük, eredményét pedig *fókuszált formának.*

Az F gráfot G-re fókuszált formának nevezzük, ha részgráfként tartalmazza G-t, G-n kívüli elemei egy formának megfelelően vannak címkézve és G-n kívül megfelel a forma feltételeinek, azaz rajta kívül nincsenek „hasonló csúcsok" és „párhuzamos élek". A fogalom pontos meghatározása a függelék 31. definíciójában olvasható. Egy G-re fókuszált forma G-beli elemeit konkrét elemeknek, G-n kívülieket absztraktnak nevezzük.

A *fókusz* műveletével elérhetjük formánk egy morfizmus mentén történő kibontását. Egy G és S közötti absztrakt morfizmus mentén történő fókusz során feladatunk egy olyan G-re fókuszált F forma készítése, ami előállhat a formánk részének konkretizálásával. A fókusz precíz leírása a függelék 32. definíciójában olvasható.

Formánk minden konkretizáltjának minden illeszkedése megőrződik a fókuszálás során, így nem hagyunk ki egyetlen konkrét morfizmust sem.



5.5. ábra. *A 5.3 ábrán látható absztrakt morfizmus szerinti fókusz. A konkrét gráfelemeket bekeretezés jelöli.*

---

14. Példa Példánkban végrehajtottuk az absztrakt morfizmus szerinti fókuszálást. A fókuszált függvényünk a 5.5 ábrán látható. Mivel az `S0` gráfban lévő `n1` csúcsnak 1 a multiplicitása, és kompatibilitási reprezentációja szerint `List` címkéjű éleket tartalmaz, összeegyeztethető vele a fókuszált `F` gráfban található `l1` csúcs. Hasonlóképpen vezethető le `c1` és `f1` helytállósága is.

A fókuszálás művelete felvet egy újabb feladatot:

Illeszkedések felsorolása: Egy következtető rendszernek meg kell adnia egy olyan (lehetőleg minimális) fókuszált gráfhalmazt, amelyben egy adott absztrakt morfizmusnak összes lehetséges fókuszáltja szerepel.

5.3.3. Szűrés

Mivel fókuszáláskor csakis azt vizsgáljuk, milyen címkéjű csúcsokat és éleket hozhatunk létre és hányat, ezért előfordul, hogy olyan fókuszált formák jönnek létre, amelyek ellentmondanak a reprezentációikban hordozott információknak. A transzformátor pontosítása végett fontos, hogy minden olyan fókuszált formát kizárjunk, amelyről belátható, hogy nem tartozhat hozzá példány.

Megeshet, hogy bár fókuszált gráfunkhoz tartoznak konkretizáltak, azok egyike sem felel meg az esetleges jólformáltsági feltételünknek. Az ilyen lehetőségeket szintén el kell vetnünk.

---

15. Példa Az illeszkedésünk fókuszálásakor előfordulhatott volna az az eset, amikor `f1` élünkkel párhuzamosan megjelenik egy szintén `first` címkéjű absztrakt él. Mivel az absztrakt él is legalább egy konkrét élt jelöl, ez az eset ellentmond a párhuzamos éleket tiltó jólformáltsági feltételünknek, ezért elvetendő.

Így egy újabb feladatot kell elvégeznünk:

Reprezentációk vizsgálata: Egy következtető rendszernek fókuszált formákról kell eldöntenie, hogy fókuszált formánk teljesíti-e a reprezentációkban lévő feltételeket. Minél több olyan esetet ki szeretnénk szűrni, ahol ez nem történik meg.

5.3.4. Transzformációs lépés

A L-re fókuszált F gráfunkon hajtsuk végre a $P = (L, R)$ gráftranszformációt. Ekkor egy új, R-re fókuszált F_1 gráfot kapunk, amelyben a reprezentációk elavultak lehetnek. A használt ekvivalenciák ismeretében meg kell állapítanunk, mely reprezentációk változhattak meg. Az összes lehetséges változást meg kell állapítanunk, fel kell jegyeznünk és tovább kell számolnunk velük.

---

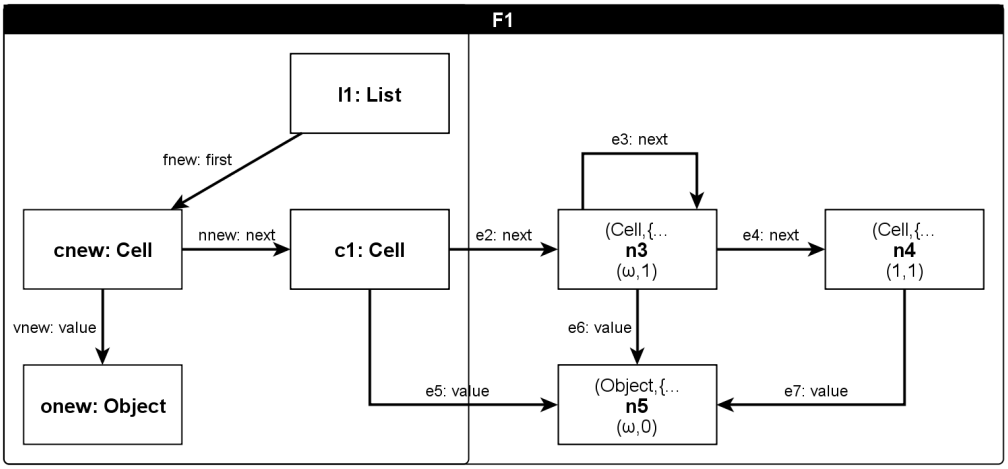
16. Példa Ha alkalmazzuk fókuszált `F` gráfunkon a `Put` transzformációt, a 5.6 ábrán látható `R`-re fókuszált gráfot kapjuk. Nézzük végig az absztrakt elemek reprezentációit, melyek változhattak.

A szomszédsági reprezentációkat vizsgáljuk először. Az `n3` csúcs a kompatibilitási reprezentációjában azt tárolja, hogy `next` címkéjű bejövő élen `Cell` címkéjű csúccsal egy alkalommal volt szomszédos. Mivel az átalakítás során sem `c1` nem változott, sem más szomszédja, ezért $comp_{\texttt{F1}}(\texttt{c3})$ is marad a régi. Hasonló a helyzet az `n5` csúccsal is.

Érdemes észrevenni, hogy 1 távolságú szomszédsági ekvivalencia esetén csak akkor módosulhat a reprezentáció, ha a formabeli csúccsal közvetlen szomszédságban lévő elemek módosulnak. Ezek szerint `n4` csúcs kompatibilitási reprezentációja sem változott.

Az instrumentális ekvivalenciáknál a multiplicitások nem változhatnak. Más a helyzet a predikátumainkkal. A `first`-tel címkézett élünk megszűnésével és egy új beszúrásával a *reachable* csúcspredikátumok teljesülését felül kell vizsgálnunk. Tudjuk, hogy amikor még a `c1` csúcsba vezetett az első csúcsot jelző él, akkor teljesült a feltétel `n3`-ra és `n4`-re is. Kikövetkeztethető, hogy ez a tulajdonság akkor is megmarad, ha `c1` a második cella. Mivel az `n5` csúcsba nem megy `next` címkéjű él, a predikátum erre nem teljesülhet.

Az absztrakt élek reprezentációi az egyszerű $\overset{\forall}{\sim}$ ekvivalenciából kifolyólag sosem változnak, így elképzelhető egy olyan transzformációs lépés, amelyben gráfunk címkéi nem módosulnak.



5.6. ábra. *A* `Put` *transzformáció alkalmazása után létrejövő* `R`*-re fókuszált forma.*

Következtető rendszerünknek meg kell tudnia oldani a következő feladatokat:

Reprezentációk átírása: Mivel a transzformációs lépés fókuszált formákra mért hatása nem mindig lokális, el kell tudnunk dönteni, mely reprezentációk változhattak meg, és hogyan. A változásokat felülről kell becsülnünk, de törekednünk kell arra, hogy minél kevesebb téves lehetőséget jegyezzünk fel.

A transzformációs lépés végrehajtása után érdemes újra elvégezni a „reprezentációk szűrése" lépést, így vetve el újabb hibás lehetőségeket.

5.3.5. Normalizálás

Normalizálás során az inverz fókuszálás műveletét hajtjuk végre, így térve vissza a formák tartományába. A művelet során először meghatározzuk a konkrét csúcsokkal összeegyeztethető reprezentációkat. Ez a művelet úgy hajtandó végre, mint a formázás, azzal a nehezítéssel, hogy a már meglévő absztrakt elemeket is bele kell venni a számításba.

A megegyező reprezentációjú elemek, és az azok között húzódó párhuzamos élek csoportokat alkotnak. A csoportok instrumentális ekvivalenciaosztályának összeegyeztethetőnek kell lennie a csoport elemeivel. Feladatunk szintén hasonló a formázáshoz, azzal a kivétellel, hogy absztrakt gráfelemek instrumentális ekvivalenciaosztályait is vizsgálni kell.

17. Példa Határozzuk meg az `F1` `R`-re fókuszált gráfunk konkrét csúcsainak kompatibilitási reprezentációit a szomszédsági ekvivalencia reprezentáló függvényével!

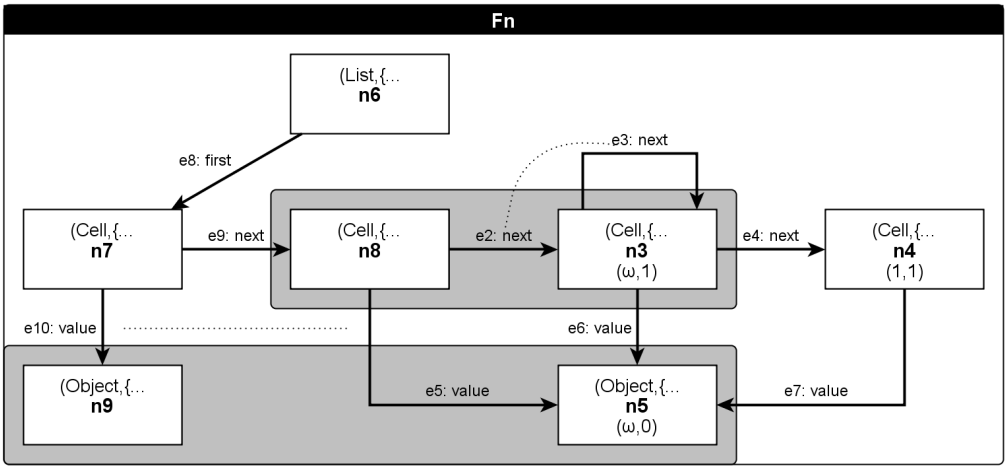
$$
\begin{array}{rcl}
nrep_{1,1}(\texttt{l1}) & = & (\texttt{List}, \{\}, \{(\texttt{Cell}, \texttt{first}, 1)\}) \\
nrep_{1,1}(\texttt{cnew}) & = & (\texttt{Cell}, \{(\texttt{List}, \texttt{first}, 1)\}, \{(\texttt{Cell}, \texttt{next}, 1), (\texttt{Object}, \texttt{value}, 1)\}) \\
nrep_{1,1}(\texttt{c1}) & = & \begin{cases} (\texttt{Cell}, \{(\texttt{Cell}, \texttt{next}, 1)\}, \{(\texttt{Cell}, \texttt{next}, 1), (\texttt{Object}, \texttt{value}, 1)\}) \\ \text{vagy} \\ (\texttt{Cell}, \{(\texttt{Cell}, \texttt{next}, 1)\}, \{(\texttt{Cell}, \texttt{next}, \omega), (\texttt{Object}, \texttt{value}, 1)\}) \end{cases} \\
nrep_{1,1}(\texttt{onew}) & = & (\texttt{Object}, \{(\texttt{Cell}, \texttt{value}, 1)\}, \{\})
\end{array}
$$

Ahogy látható, konkrét szomszédság esetén ugyanazt a műveletet kell végrehajtani, mint formázáskor. Gondot jelentett viszont a `c1` kiszámítása, mert bár tudtuk, hogy kimenő élen szomszédos egy másik `Cell`-lel, nem tudtuk a szomszédság párhuzamosságát megállapítani, mert éleink nem hordoznak semmilyen adatot. Ekkor az összes lehetőséget fel kell venni, így lesz olyan esetünk, amikor a párhuzamosság 1, és lesz, amikor ω. A példában az 1-es a lehetőséget számoljuk végig.

A 5.7 ábrán láthatóak az ugyanolyan kompatibilitási reprezentációval rendelkező csoportok. Az instrumentális ekvivalenciaosztályok megítélése a csak konkrét elemekből álló csoportok esetén ugyanúgy zajlik, mint fókuszáláskor, ezért `n6` és `n7` címkéi egyértelműek. Multiplicitások megítélésénél fel kell fedezzük, hogy egy egyelemű és egy ω elemű halmaz uniójának számossága ω. Ezzel megállapítottuk az összevont csoportok multiplicitását. A csoportokat leíró predikátumok összevonásánál sem kerülünk nagy bajba, hiszen P-vel és Q-val reprezentált csoport uniójának reprezentációja $P \cup Q$. Ebből kifolyólag a középső cellák elérhetősége $1 \cup 1 = 1$, az objektumoké pedig $0 \cup 0 = 0$.

Élek esetén az összevonás az egyosztályú ekvivalencia miatt egyértelmű.

Ha elvégezzük az összevonást, az `SO`-val izomorf formát kapunk.



5.7. ábra. *Kompatibilitási reprezentáció megállapítása utáni csoportokat bemutató ábra. Az egynél többelemű csúcscsoportok bekeretezéssel vannak összerendelve, az így kialakuló párhuzamos éleket pontozott vonal köti össze.*

Újabb kihívást találtunk következtető rendszerünknek:

Reprezentációk számítása: Egy következtető rendszernek meg kell tudnia adnia a lehetséges kimenetelét a reprezentáló függvényeknek, hogy kiderítse a lehetséges reprezentációkat.

$\exists$ **Gráftranszformáció**	$\Longrightarrow$	$\exists$ **Formatranszformáció**
$\Downarrow$		$\Downarrow$
Gráf elérhető	$\Longrightarrow$	**Forma elérhető**
$\Downarrow$		$\Downarrow$
Gráf szerepel az állapottérben	$\Longrightarrow$	**Forma szerepel az absztrakt állapottérben**

5.1. táblázat. *Gráfok és formák viszonya az állapottérben*

5.3.6. Konzisztenciavizsgálat

A transzformáció végrehajtása végén elvégezzük a forma konzisztenciavizsgálatát is:

Konzisztenciavizsgálat: Egy következtető rendszernek ki kell derítenie, egy formának lehet-e konkretizáltja.

Mivel minden lépésnél vigyáztunk arra, hogy felülről becsüljük meg a lehetséges átmeneteket, a normalizálás végén minden lehetséges absztrakt állapotátmenetet megkapunk. Mindemellett a formák részleges kifejtésénél sok hamis állapotátmenetet kizártunk.

5.4. Absztrakt állapottér

A kiindulási S_0 formából t absztrakt transzformátor R gráftranszformációs szabályait úgy alkalmazzuk, hogy ne kerüljünk olyan állapotba, amely megsérti P-beli jólformáltsági kritériumainkat, absztrakt állapotteret épít. Ezt ugyanúgy ábrázoljuk, mit a szokványos, gráfokból épülőt, azzal a különbséggel, hogy a csúcscímkék formák. Az elérhetőség és az állapottér formális meghatározásai a függelék 33. és 34. definíciójában szerepelnek.

Az absztrakt állapottér vizsgálhatóságát mondja ki az alábbi állítás:

1. Állítás (Absztrakt állapottér mérete). *Az absztrakt állapotterünket ábrázoló gráf mérete* ***véges****.*

Mivel az adott címkekészlettel címkézett formák száma és a gráftranszformációk száma véges, állapotterünk címkekészletei véges halmazok. Mivel az absztrakt állapottér címkéi injektíven vannak hozzárendelve egy véges halmazhoz, az absztrakt állapotterünk csúcsainak száma is véges. Két állapot között maximum $|R|$ állapotátmenet lehet, így állapotterünk éleinek száma is véges.

Mivel állapotterünk véges, így felépíthető és vizsgálható. A 5.1 táblázatban megfigyelhetjük a gráfokon és a formákon értelmezett fogalmak és műveletek összefüggéseit.

Ha egy következtetővel be tudjuk látni, hogy állapotterünk minden formájának konkretizáltja teljesíti a P_S biztonsági kritériumot, belátható hogy gráfnyelvtanunk biztonságos.

Biztonság ellenőrzése: Egy következtető rendszernek ellenőriznie kell, hogy minden formánk teljesíti a biztonsági kritériumot.

5.5. Kapcsolódó munkák

Az általam kidolgozott absztrakt transzformátor a kapcsolódó irodalmakban [11, 14, 8] alkalmazott elgondolásokon alapszik, egyben általánosítja és összefoglalja azokat. A 5.2. táblázat bemutatja, mely lépések milyen fogalmaknak feleltethetőek meg.

Lépés	Predikátumformák	Szomszédsági formák
1. Absztrakt illesztés	-	pre-matching
2. Fókusz	focus	materilalisation
3. Szűrés	coerce	-
4. Transzformációs lépés	transformation + coerce	concrete shape transformation
5. Normalizálás	embedding	normalisation
6. Konzisztenciavizsgálat	(coerce-ben)	-

5.2. táblázat. *Kapcsolódó munkák*

Predikátumformák eredeti interpretációjukban háromértékű logikai struktúrákként vannak megvalósítva. Ezeken úgynevezett „*predicate update formulae*"-kat hajtunk végre. A művelet precizitásának növelése érdekében először megvizsgáljuk az összes lehetséges kétértékű interpretációját struktúránk azon részének, melyen a transzformációt végrehajtanánk; ezt a műveletet nevezzük *focus*-nak, mely megfeleltethető az itt definiált fókusz lépésnek.

A focus lépés után megszűrjük azokat a lehetőségeket, melyek ellentmondásra vezetnének. Ezt a műveletet *coerce*-ként nevezik. A kétértékű részstruktúrán ezután végrehajtjuk magát a transzformációt. Több hibás eredmény kiszűrése céljából újra elvégezzük a coerce műveletet, majd egy *embedding* nevű eljárással újra háromértékűvé alakítjuk struktúránkat.

Predikátumformák konzisztenciavizsgálata fókuszált állapotban történik, a coerce funkció esetek elvetésére is szolgál.

Szomszédsági formák esetén a „*pre-matching*" lépés absztrakt illesztésnek, „*materilalisation*" művelet pedig fókusznak feleltethető meg. Mivel multiplicitásokkal történő számolás pontos és egyserű művelet, külün szűrést nem alkalmaznak. Transzformációs lépés után szomszédsági csúcsok reprezentációiból állapítjuk meg a konkrét elemek szomszédságát, majd a „*normalisation*" lépéssel térünk vissza az absztrakt gráfok terére.

Szomszédsági formákkal kapcsolatos irodalomban nem merült fel konzisztenciavizsgálat módszere, egyedül [8] cikk Conjucture 33. pontjában szerepel egy sejtés, miszerint a kérdés szomszédsági formákra általánosságban eldölthetetlen.

5.6. Összefoglalás

A fejezetben megadtam a transzformációs lépés szükséges feltételét, majd egy ezt teljesítő, általam kidolgozott absztrakt transzformátort. A transzformátor az eddigi absztrakciós technikákban egységesen alkalmazott módszert követi, miszerint formánkra illesztjük transzformációs szabályunk baloldalát, majd az illesztés mentén „kibontjuk" absztrakt gráfunk. A félig absztrakt, félig konkrét gráfunk fókuszált gráfnak neveztük, konkrét részén a megszokott módon végre tudtuk hajtani a transzformációs lépés. A művelet után fókuszált formánkat „visszacsomagoljuk" rendes, absztrakt formává.

Az absztrakt transzformáció lépései több helyen nem determinisztikusan vannak megfogalmazva, úgynevezett következtetési feladatok eredményei vezérlik az esetszétválasztást. Ezen feladatokra a következő fejezet tér ki.

6. fejezet

Következtetés architektúrája

6.1. Következtetési feladatok

Foglaljuk össze, milyen következtetési feladatokat kell elvégeznünk!

1. **Konkrét illesztés:** Feladatunk egy minta összes lehetséges absztrakt morfizmusának felsorolása. Egy illeszkedés biztosan helytelen, ha be tudjuk látni, nem létezhetnek olyan csúcsok és élek, amelyekre:

   - teljesülnek a forma reprezentációiból származó feltételek, és
   - teljesülnek a minta szerkezetéből származó feltételek.

   Például egy olyan csúcsra nem illeszthetünk `Cell` címkéjű csúcsot, amelyre teljesül az *isList* feltétel, hiszen a két különböző címke ellentmond egymásnak.

2. **Illeszkedések felsorolása:** Feladatunk egy absztrakt morfizmusnak megfelelő összes fókuszált forma előállítása. Fókuszáláskor adottak a konkrét elemek, és nem módosulnak az absztrakt morfizmusban nem érintett absztrakt elemek sem. Megmaradó absztrakt elemek kompatibilitási reprezentációi sem változhatnak. Ezek szerint a következő lehetőségeket kell megvizsgálnunk:

   - megcáfolható-e az illesztés az instrumentális reprezentációkban tárolt információkkal,
   - eltűnhet-e érintett absztrakt gráfelem, és
   - ha nem tűnik el, hogy módosulhat instrumentális ekvivalenciája

   Például ha egy csúcs multiplicitása 2, akkor egy elem „kibontásakor” csakis 1-re csökkenhet ennek értéke, két elem kibontásakor csakis eltűnhet csúcsunk, három esetén meg ellentmondásra jutunk.

3. **Reprezentációk vizsgálata:** Feladatunk fókuszált formánkat megvizsgálni, nem hoztunk-e létre olyan esetet, ami biztosan nem fordulhat elő. Az ilyen eseteket igyekszünk eldobni, úgy, hogy megpróbáljuk belátni: van olyan reprezentáció formánkban, amit nem adhatott eredményül reprezentáló függvény. Például előfordulhat, egy csúcs

reprezentációja meghatározza szomszédjai számát, és úgy fókuszáljuk formánkat, ami ellentmond ennek a mennyiségnek.

4. **Reprezentációk átírása:** Mivel a transzformációs lépés fókuszált formákra mért hatása nem mindig lokális, el kell tudnunk dönteni, mely reprezentációk változhattak meg, és hogyan. Például egy szomszédsági ekvivalencia reprezentációja nem változhat meg, ha a változás kellően távol (szomszédság sugaránál távolabb) történik tőle.

5. **Reprezentációk számítása:** Feladatunk fókuszált formából rendes formát készíteni. Ezt úgy tesszük meg, hogy kiszámítjuk a konkrét elemek reprezentációit az ekvivalenciák reprezentáló függvényeinek segítségével. Azonban reprezentáló függvények gráfokon vannak értelmezve, így nehézségekbe ütközünk az eredmény kiértékelésekor. Ebben a következtető lépésben fel kell tudnunk sorolni a konkrét elemek lehetséges kompatibilitási reprezentációit. Miután megkaptuk a lehetőségeket, a kompatibilitás szerint hasonló elemeket összehúzzuk, majd ezeknek megvizsgáljuk a lehetséges instrumentális reprezentációit.

6. **Konzisztenciavizsgálat:** Egy transzformáció eredménye minden igyekezetünk ellenére tartalmazhat ellentmondást. Ilyen esetben formánknak biztosan nem lehet konkretizáltja, így az ilyen állapotok olyan tartományokba vihetik el állapotterünket, amelyekbe konkrét állapotterünk nem jutna el. Ha egy következtető rendszer be tudja látni egy formáról, hogy inkonzisztens, akkor azzal javíthatjuk analízisünk sikerességének esélyeit.

7. **Biztonság ellenőrzése:** Egy következtető rendszernek ellenőriznie kell, hogy egy forma biztosan teljesíti-e az adott biztonsági kritériumot. Ha akad absztrakt állapotterünben olyan elem, amelyre nem tudjuk ezt belátni, helyességellenőrzésünk meghiúsul.

Amint láthatjuk, több eltérő karakterisztikájú következtetési feladatot kell elvégeznünk az absztrakt transzformáció végrehajtásához.

6.2. Architektúra

6.2.1. Követelmények

Az ekvivalencia alapú technika definiálásakor az absztrakció általánosíthatósága és a bővíthetősége volt a legfőbb cél. Ennek következtében a transzformáció lépéseiben nem pontos szabályok, hanem következtetési feladatok szerepelnek. A probléma természetéből fakadóan feladatosztályonként és absztrakciónként is különböző tulajdonságokkal rendelkező következtető rendszerek alkalmazása lenne az ideális.

Ennek elérése céljából olyan architektúrára van szükség, amely az alábbi feltételeket teljesíti:

1. Bővíthető legyen következtető rendszerekkel.

2. Ha egy feladatra több következtető is alkalmas, a végeredmények pontosítsák egymást.

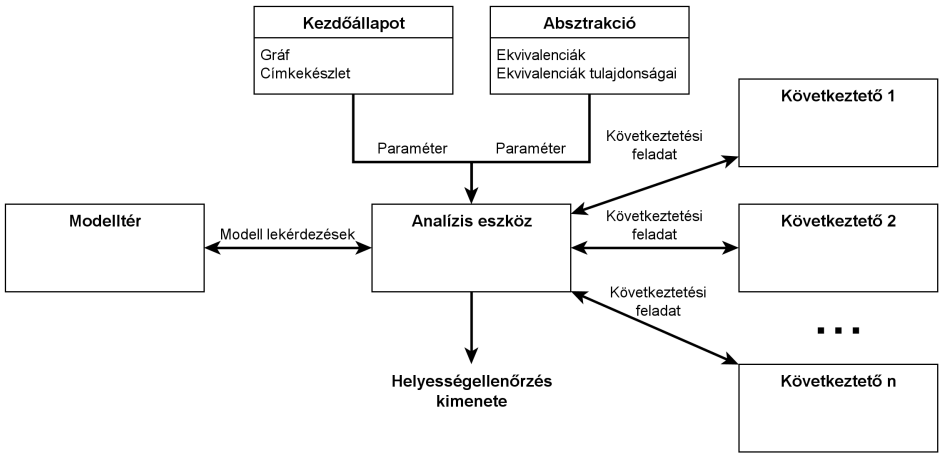
3. Ha egy feladat egy következtetővel sem oldható meg, az csak a pontosságot csökkentse, de ne akadályozza a végrehajtást.

Az 1. pont lehetővé teszi, hogy több, eltérő karakterisztikájú következtető rendszer is tudjon dolgozni az adott problémán. Így elérhetővé válik, hogy absztrakciókkal bővíthető legyen rendszerünk, hiszen új ekvivalencia mellé szükség szerint új következtető adható. Az analízis eszköz az absztrakció függvényében oszthatja el a feladatokat ezek között.

A 2. pont hatására nagyban javítható a következtetés pontossága és hatékonysága. Elérhető, hogy olyan problémák is megoldhatóak legyenek, amelyekre külön-külön egyik megoldásunk sem alkalmas. Ha ismerünk egy pontatlan de hatékony és egy pontos de költséges megoldást, megvalósítható, hogy a hatékony eljárás elvégezze a feladatok nagy részét, és a pontosnak csak a maradékot kelljen ellenőriznie. Ez főképp olyan feladatokban fontos, ahol a költséges megoldás alkalmazása kivitelezhetetlen lenne az összes lehetőségen.

A 3. pont szerint olyan esetekben is végrehajthatónak kell lennie az analízisnek, ha nem tudjuk a következtetést elvégezni. Ennek oka lehet, hogy nem létezik vagy nem ismerünk konkrét feladatra következtetést, esetleg túl nagy erőforrást (például elfogadhatatlanul sok időt) igényelne annak végrehajtására.

6.2.2. Felépítés



6.1. ábra. *Architekturális felépítés.*

A rendszer felépítése a 6.1 ábrán látható.

Analízis eszközünk paraméterként kap egy kezdőállapotot, amely tartalmazza a kiindulási gráfot és annak címkekészletét. Ezen kívül ekvivalenciák formájában definiálják a használni kívánt absztrakciót is, amivel együtt megérkeznek a reprezentáló függvények és a következtetési sajátosságok is.

Az analízis eszköz egy modelltéren dolgozik. Ebben tárolódnak az állapotok, és itt találhatóak a számítások részeredményei is. A modelltér adatai lekérdezésekkel érhetőek el.

Eszközünk képes több, szolgáltatásként megjelenő következtető rendszernek feladatokat osztani majd eredményeket fogadni tőlük. A következtetők az analízis eszközön keresztül elérhetik a feldolgozás alatt álló állapotot, és átalakíthatják egy külső eszköz bemenetének megfelelő formátumúvá.

A folyamat végén a helyességellenőrzés kimenetelét, az állapotteret valamint a művelet során alkalmazott levezetéseket kimenetként megkapjuk.

6.2.3. Modelltér

Analízis eszköz modelltereként a Viatra2[15] modelltranszformációs rendszert használom. Ez az Eclipse alapú technológia alkalmas nagy méretű és kifejezőerejű modellterek tárolására, és valamint a modellelemek hatékony lekérdezésére.

6.2.4. Következtető rendszerek

Analízisünk következtetési feladatait keretrendszeren kívüli szoftverkomponensek hajtják végre. A következtető rendszerek bemenetként megkapják a következtetési feladatot, elérhetik végrehajtás alatt álló formánkat vagy fókuszált formánkat, illetve kaphatnak opcionálisan kaphatnak megoldási lehetőségeket. Mivel minden következtetési feladat csak véges sok lehetséges lépést tartalmazhat, a lehetőségek igény szerint generálhatóak és felsorolhatóak. A következtetési feladat eredményeként lehetséges eredmények halmazát kapjuk meg.

Ahogy az absztrakciók, a következtetési feladatot végrehajtó rendszerek is igen sokfélék lehetnek. A dolgozat során négy ilyen komponenst vizsgálok meg, hogy melyek, milyen absztrakciók mellett milyen hatékonysággal és pontossággal tudják megoldani az adott feladatot. Tapasztalataimat a 6.1. táblázat ábrázolja.

6.3. Összefoglalás

A fejezetben felsoroltam és pontosítottam az absztrakt transzformáció lépéseiben felmerülő következtetési feladatokat. Megfogalmaztam az ekvivalencia alapú analízis megvalósítását célzó, következtető rendszerekkel erősen bővíthető architektúra követelményeit, majd megvalósítási javaslatot tettem rá. Végül definiáltam egy felületet, melyen keresztül a keretrendszeren kívüli következtető rendszerek megvalósíthatják feladataikat.

A fejezet végén összefoglaló táblázatba foglaltam az általam megvizsgált következtető rendszerekkel kapcsolatos tapasztalataimat.

	Abox lekérdezés	Gráfminta-illesztés	Multiplicitás algebra	Tételbizonyító
Konkrét illesztés	esetfüggő/szűrés	hatékony/szűrés	-	drága/pontos
Illeszkedések felsorolása	-	-	hatékony/esetfüggő	drága/pontos
Reprezentációk vizsgálata	-	hatéony/szűrés	-	drága/pontos
Reprezentációk átírása	esetfüggő/esetfüggő	hatékony/szűrés	-	drága/pontos
Reprezentációk számítása	esetfüggő/szűrés	hatékony/szűrés	hatékony/esetfüggő	drága/pontos
Konzisztenciavizsgálat	-	-	-	drága/szűrés
Biztonság ellenőrzése	-	-	-	drága/pontos

Jelölés: a következtetési feladatok hatékonyság/pontosság szerint lettek osztályozva, ahol:

- : Nem ismert használható alkalmazás.
hatékony : A következtetési feladat várhatóan könnyen elvégezhető.
drága : A számítási költség miatt megvalósíthatósági akadályokba ütközhetünk.
szűrés : A következtető rendszer gyakran nem tud pontos eredményt adni, de sok esetet ki tud zárni.
pontos : A következtető rendszertől azt várjuk el, hogy legtöbbször pontos válasszal szolgáljon.
esetfüggő : A következtető rendszer teljesítménye nagyban függ a használt absztrakciótól.

6.1. táblázat. *Következtető rendszerek alkalmazhatósága*

7. fejezet

Példányszintű következtető rendszerek

Példányszintű következtető rendszerek alatt olyan következtetési szabályokat értünk, melyek a formákat gráfként kezelik, és a formázás során megmaradt invariáns tulajdonságokat kívánják kihasználni.

7.1. Mintaillesztés formákon

Formáink éleinek *olab* reprezentációi egyértelműen és közvetlenül meghatározzák, milyen címkékkel egyeztethető össze az absztrakt él. Vizsgáljuk meg formánk csúcsait is, melyek milyen címkéjű elemeket rejthetnek.

Vezessünk be feldolgozás alatt álló S forma csúcsain egy új címkézést, amely minden csúcshoz a vele összeegyeztethető címkehalmazt rendeli: $complab_S : N_S \mapsto 2^{Lab^N}$. Ezt a címkézést többféleképpen is előállíthatjuk:

1. Következtetési feladatként megvizsgáljuk, mely kompatibilitási reprezentációk mely csúcscímkékkel egyeztethetőek össze. A következtetést ezek után elvégezzük az instrumentális reprezentációkkal is. A címkehalmazt a kettő metszete határozza meg.

2. Következtetési feladatként S forma minden csúcsára megvizsgáljuk: belátható-e hogy egy adott címke nem egyeztethető össze a csúccsal. A ki nem zárt címkék halmaza alkotja a címkézést.

Az első módszer előnye, hogy a számítás könnyen, akár előre is elvégezhető, a másodiké hogy pontos. A következtetési feladat visszavezethető egy egycsúcsú gráf „konkrét illesztésére".

Ha van egy G minta gráfunk, illeszthetjük azt egy olyan m *címketartalmazó morfizmus* szerint S formánkra, hogy az alábbi feltételeket betartsa:

1. Minden $n \in N_G$ csúcsra teljesül, hogy $lab^N_G(n) \in complab_S(m(n))$, azaz a csúcs címkéje szerepel a forma csúcsával kompatibilis címkék közt.

2. Minden $e \in E_G$ élre teljesül, hogy $lab^E_G(e) = olab_S(m(e))$, azaz az élcímkék is összeegyeztethetőek.

Az ilyen morfizmus abban különbözik a megszokott illesztéstől, hogy nem szükségszerűen injektív, és csúcscímkéknél nem egyenlőséget, hanem tartalmazást vizsgál.

Ha G_1-ből vezet m címkehelyes morfizmus G_2-be, és G_2 formázása során m_f formázási morfizmussal előáll S_2 forma, akkor az $m_f \circ m$ egy címketartalmazó morfizmus lesz. Ez az összefüggés felhasználható arra, hogy felülről becsüljünk lehetséges illeszkedéseket, szomszédságokat és olyan csúcspredikátumokat, melyek nem tartalmaznak tagadást. Ezek szerint a címketartalmazó morfizmussal történő illeszkedések eleve kielégítik a „konkrét illesztések" keresésére irányuló következtetés feladatot.

7.2. Abox lekérdezések

7.2.1. Leíró logika

A *leíró logika* egy szakterülethez tartozó fogalmi rendszer leírására kifejlesztett formalizmus [4]. Ismereteinket úgynevezett tudásbázisban tároljuk, amiből következtetéseket és lekérdezéseket hajthatunk végre. Tudásbázisunk két fő komponensre tagolható: Tbox és Abox. A Tbox tartalmazza a fogalmi rendszer axiómáit, vagy más néven *koncepcióit*. A koncepciók mellett konkrét adatokról, azaz koncepciók *példányairól* is számon tartunk információkat, melyek az Abox komponensben találhatóak. Ebben a fejezetben ezen a komponensen fogunk következtetni.

Az Abox kétféle kifejezést tartalmazhat, mindkettőből tetszőleges mennyiséget:

1. `C(a)`: az `a` egyed eleme a `C` koncepciónak.

2. `R(a, b)`: az `a` és a `b` egyed között vezet `R` reláció.

A kifejezéseknél alkalmazzuk az úgynevezett egyedinév-kikötést (angolul: unique name assumption), miszerint tetszőleges $\texttt{a} \neq \texttt{b}$ változók esetén `a` és `b` különböző egyedeket jelöl.

7.2.2. Formák leképezése Abox-ba

Hogy következtetéseket tudjunk formáinkból vonni, először le kell képeznünk őket leíró logikai kifejezésekre. Ebben a fejezetben Abox-ban kívánjuk formáinkat tárolni.

Legyen Lab^N a csúcscímkék halmaza. Ezekből a címkékből egyszerű koncepciókat képzünk, jelöljük őket `lab_N_` prefixszel, például a `List` címkéből `lab_N_List` koncepció képződik. Hasonló képpen Lab^E élcímkékből is képezzünk relációkat, melyeket `lab_E_` prefixszel jelöljünk. Tehát a `first` élcímkét `lab_E_first` relációval szeretnénk ábrázolni.

A címkekészletnek megfelelő gráfokból egy S formát állítunk elő. Tudásbázis egyedei legyenek formánk csúcsai, jelöljük őket az alábbi változókkal: `n1`, `n2` ... A változók között relációk vezethetnek. Pontosan akkor teljesüljön egy *lab* címkének megfelelő `lab_E_lab` reláció egy indivídum-párra, ha formánkban a két csúcs között vezet olyan e él, melyre $olab_S(e) = lab$.

Konkrét illesztések előállítása érdekében az egyedekhez hozzá kell rendelni, mely címkékkel összeegyeztethetőek. Ennek érdekében minden egyedhez meg kell adni: mely egyszerű,

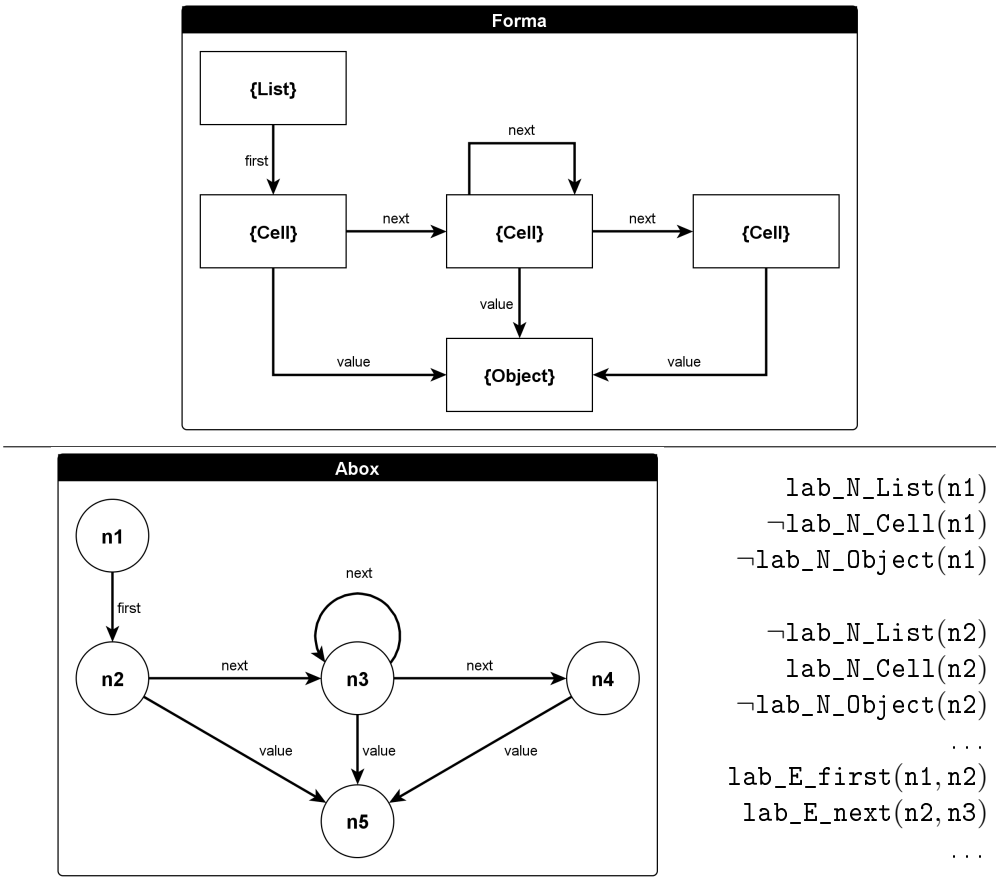
címkét jelző koncepciók teljesülnek rá, és melyek azok, amik nem. Ha egy `C` koncepció nem teljesül `a` egyedre, azt megadhatjuk úgy, hogy `a` a $\neg$`C` példánya.

---

18. Példa Tekintsük meg a 7.1. ábra felső részén szereplő felcímkézett formát példaként. A forma minden csúcsához egyelemű címkehalmaz tartozik, azaz egyértelmű mely címkéjű konkrét csúcsok egyeztethetőek velük össze. Ezt a formát szeretnénk leképezni tudásbázisra, mely az ábra alsó részén látható.

Először határozzuk meg a tudásbázis egyedeit: `n1`, ..., `n5`. Másodszor fogalmazzuk meg, mely egyedekhez mely koncepciók tartozhatnak: egyesével felírjuk minden egyedre az összes olyan címkézést jelző koncepciót, amely címkézés összeegyeztethető a forma csúcsával, és negálva azokat, amelyek nem. Például `n1` egyedre `lab_N_List` teljesül, de `lab_N_Cell` nem. Az `n1` és `n2` egyedre felírt címkézés megtekinthető az ábra alsó részének jobb oldalán.

A példányok között haladó relációkat is meg kell határozni. Mivel formánkban az `n1`-nek és az `n2`-nek megfelelő csúcs között vezet `first` címkéjű él, az `lab_E_first(n2,n2)` állítást is fel kell venni tudásbázisunkba.



7.1. ábra. *Forma leképezése tudásbázisra*

7.2.3. Lekérdezések

Elkészült tudásbázisunkon példányszintű Abox lekérdezéseket hajtunk végre [7]. A lekérdezéseket egy következtető dolgozza fel, a kritériumnak megfelelő példányokat a számítás

után megkapjuk. A feltehető kérdéseket erősen korlátozza a szigorú lekérdezőnyelv, mely biztosítja a következtetés végrehajthatóságát.

---

19. Példa Keressük meg, formánk megy csúcsaira teljesülhet a $firstCell(n) \Leftrightarrow Cell(n) \wedge \exists u First(u, n)$ predikátum. A lekérdezés olyan koncepció példányaira irányul, melyek címkéje `Cell`, és vezet bele `first` címkéjű él. Ezekből a feltételekből egyértelműen kiolvasható az alábbi koncepció:

$$\texttt{lab\_N\_Cell} \sqcap \exists \texttt{lab\_E\_first}^{-}$$

7.2.4. Értékelés

Leíró logikák példányszintű lekérdezéseit nagyban befolyásolja a gyenge lekérdezőnyelv. Illesztendő mintánk összes lehetséges illeszkedését felülről becsülhetjük ontológiai lekérdezésekkel, de pontatlansága és gyenge hatékonysága miatt ilyen funkcióra nem érdemes ilyen célokra használni az eszközt.

A lekérdezések alkalmasak viszont szomszédságok és egyszerűbb, negatív mintát nem tartalmazó predikátumok felső becslésére.

7.3. Gráfmintaillesztési technikák

7.3.1. Mintaillesztés

Gráf alapú modellek fölötti lekérdezéseket *gráfmintaillesztésnek* nevezzük. A lekérdezés során egy lekérdezőnyelvnek megfelelő formájú *mintát* adunk meg, melyben meghatározhatjuk a lekérdezésben szereplő gráfelemek tulajdonságait. A mintaillesztés eredményeként olyan gráfelemeket kapunk, melyek kielégítik a mintában megfogalmazott feltételeket.

A Viatra2 modelltranszformációs keretrendszer rendelkezik gráfmintaillesztés technikájával [5]. A rendszer inkrementális gráfmintaillesztő technikát használ, ami annyit tesz, hogy egy minta tárolja és folyamatosan karbantartja a gráfminták illeszkedéseit, nagyságrendekkel gyorsítva meg a lekérdezések megválaszolását.

7.3.2. Formák leképezése modelltérbe

A Viatra2 rendszer alap tárolási egysége a modelltér. A modellterekben gráfstruktúrába rendeződnek a modellelemek, ahol a csúcsokat entitásoknak, az éleket relációknak nevezzük. Az entitások tartalmazási hierarchiába szerveződnek, amely mentén a lokális névvel rendelkező modellelemek egyedi globális azonosítót kapnak. A modellelemek saját léttel rendelkeznek, tetszőlegesen elérhetőek, létrehozhatóak vagy törölhetőek.

Az elemek között típusosságot és öröklési viszonyt meghatározó kapcsolatok lehetnek, így a modell és a metamodell azonos módon kezelhető és egy helyen tárolható. Egy elemnek több típusa és őse lehet, ezek dinamikusan változhatnak. Egy típus összes példánya könnyen elérhető.

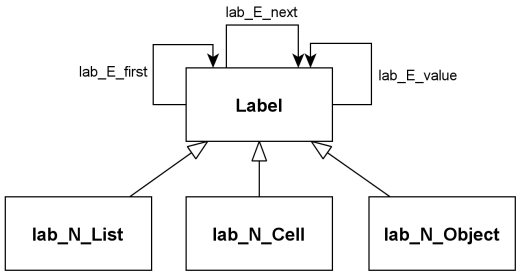
Lab^N csúcscímkéinket és Lab^E élcímkéket típusokat tároló metamodellben helyezzük el. Mivel a címkézett gráfok definíciója eredendően ezt megengedi, bármilyen címkéjű élek vezethetnek bármilyen csúcsok között. Hogy ezt a feltételt metamodellünk kielégítse, vegyünk fel egy entitást, mely tetszőleges címkét jelöl, és nevezzük `Label`-nek. Ebből származtatjuk mindegyik Lab^N-beli címkét jelölő entitásunk, melyeknek példányai lesznek a címkével összeegyeztethető csúcsok. Magának `Label`-nek nem lesz közvetlen példánya. Elnevezés terén ismét alkalmazzuk a `lab_N_` prefixumot.

Lab^E-beli élcímkéinket olyan relációkkal ábrázoljuk, melyeknek forrása és célja is a kitüntetett `Label` entitás. A relációk a `lab_E_` prefixumot viselik, példányaik lesznek a megfelelő címkével jelölt élek.

Ilyen konstrukció mellett megvalósítható, hogy tetszőleges címkéjű élek tetszőleges címkéjű csúcsok között vezethessenek. Ha jólformáltsági kritériumaink megszabják, akár szigorúbb címkézés is megvalósítható.

---

20. Példa Vegyük példaként a $Lab^N = \{\texttt{List}, \texttt{Cell}, \texttt{Object}\}$ csúcscímkékkel és a $Lab^E = \{\texttt{first}, \texttt{next}, \texttt{value}\}$ élcímkékkel jelölt gráfokat. Ezek a 7.2. ábrán látható metamodellt adják.



7.2. ábra. *Címkézést megadó metamodell.*

Formáinkat egy állapotot jelző elem névterében tároljuk. A névtérben szereplő entitások jelzik a forma csúcsait, típusai az összeegyeztethető címkéket. Éleinket reprezentációk ábrázolják, melynek *olab* címkéjét az egyetlen típusa adja meg.

7.3.3. Lekérdezések

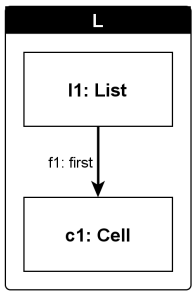
„Konkrét illesztés" feladat végrehajtásakor illesztendő gráfunk struktúrája és címkézése explicit átírható a Viatra2 lekérdezőnyelvének megfelelő bemenetre. A lekérdezés mintájában minden csúcsra megkötjük annak címke szerinti típusát, és megadjuk, hogy a feldolgozás alatt álló állapot névterében keresse az elemeket. Miután a csúcsokat változókhoz kötöttük, megadjuk az éleket, melyek a megfelelő változók között futnak.

---

21. Példa A 7.3. ábra baloldalán szereplő `L` gráfot szeretnénk formánkra illeszteni, ami a `Push` transzformációs szabály baloldalát írja le. Ebből képzünk Viatra2 mintát.

A keretrendszerbe mintáink machine-ekbe, ezek névterekbe rendeződnek. Mintánk az `AM_Push` nevet viseli, névterét az 1. sor, a metamodell névterét a 2. sor adja meg. Paraméterlistájában `State`-tel adjuk meg a vizsgálandó forma névterét, `L1`,`C1` és `F1` változókba

Illesztendő gráf



Viatra2 minta

```
1 namespace Shaping.Transformation;
2 import Modelspaces.L4.Metamodel;
3 @incremental
4 machine Push
5 {
6     shareable pattern AM_Push(State,L1,C1,F1)=
7     {
8         entity(State);
9         lab_N_List(L1) in State;
10        lab_N_Cell(C1) in State;
11        lab_N_List.lab_E_first(F1,L1,C1);
12    }
13 }
```

7.3. ábra. *Illesztendő gráf leképezése Viatra2 mintára.*

pedig a minta elemei illesztődnek. Mintánkban a helyes címkézés érdekében megkötjük az elemek típusát, és az `in State` kifejezéssel meghatározzuk, hogy a paraméterként kapott állapotban kívánunk keresni.

A 3. sorban szereplő `@incremental` annotációval adjuk meg, hogy inkrementálisan szeretnénk mintánkat illeszteni.

7.3.4. Értékelés

A Viatra2 rendszer gráfmintaillesztési technikái lekérdezőnyelv kifejezőerejében és végrehajtás hatékonyságában is jobb eredményeket mutat az Abox lekérdezéseknél.

A Viatra2 rendszer alkalmas a „konkrét illesztések” megtalálására, és a tetszőleges elsőrendű predikátum teljesülését képes ellenőrizni. Mivel bonyolult lekérdezéseket is gyorsan meg képes válaszolni, érdemes modellterünkbe minél több információt elhelyezni formánkról, így bonyolultabb lekérdezésekkel pontosabb eredményt kaphatunk.

7.4. Multiplicitás algebra

Tipikus következtető rendszerek körülményesen és alacsony hatékonysággal kezelik az olyan feladatokat, melyekben egy halmaz méretét kell kiszámítaniuk. Multiplicitásoknál ezt a következtetést egyszerű számítással is el tudjuk végezni, így mindenféleképpen érdemes ezt az utat követnünk.

Legyen m az absztrakció paramétereként megadott maximális multiplicitás, így értelmezési tartományunkat M_m jelöli. Értelmezzünk ezen a halmazon egy $+$ *összeadás* műveletet, úgy, hogy $mult_m(a) + mult_m(b) = mult_m(a+b)$ azonosság teljesüljön. Ez a feltétel egyértelműen azonosítja az összeadás műveletét, melyet a függelék 35. definíciója tartalmaz. Ha normalizáláskor a konkrét gráfelemeknek 1-es multiplicitást tulajdonítunk, akkor az összevont gráfelemek összes multiplicitását meghatározhatjuk a gráfelemek multiplicitásainak összegeként.

Fogalmazzunk meg továbbá egy *minus*-sal jelölt *levonás* műveletet is, amely a multiplicitások lehetséges, eggyel kisebb értékeit eredményezi. A függvénytől elvárjuk, hogy minden a multiplicitás esetén teljesítse a $a \in minus(a+1)$ feltételt. A feltétel most is egyértelműen

meghatározza a függvényt, melyet a függelék 36. definíciója ír le. Multiplicitások levonásával meg tudjuk határozni a fókuszáláskor kialakuló multiplicitásokat: egy absztrakt elem multiplicitásából annyiszor vonunk le értéket, ahány konkrét elemet fókuszálunk belőle. Ha 0 megjelenik az eredmények között, az az absztrakt gráfelem eltűnését jelenti, az üres halmaz pedig ellentmondásosságot.

Ahogy láttuk, az „Illeszkedések felsorolása" és a „Reprezentációk számítása" feladatok multiplikációt alkalmazó instrumentális ekvivalencia esetén hatékonyan végezhető el.

7.5. Értékelés

Az ebben a fejezetben leírt következtetők alkalmazásával egy hatékony számításokat végző, a formatranszformáció feladatköreire specializált következtető rendszereket kapunk. Bár általános esetben precizitásuk esetszűrésre korlátozza alkalmazásuk, a dolgozatban leírt ekvivalenciákra kielégítő működést tud felmutatatni, a [14, 8] cikkekben leírt eljárások is csak ilyen lépéseket használnak.

7.6. Összefoglalás

Ebben a fejezetben megadtam a címketartalmazó morfizmus fogalmát, amelynek segítségével hatékonyan szűrhetjük a következtetés lehetőségeit. A címketartalmazó morfizmus azonosságai ismertek a szakirodalomban, hasonló eredményt mutat a [11] cikk „Embedding Theorem" egy speciális, valamint [14, 8]-ben található „pre-matching" általánosított megfogalmazása.

Következtetési feladatok elvégzése céljából megadtam egy-egy általam kidolgozott leképezést, mely segítségével egy formáról ontológiai lekérdezések és gráfmintaillesztési technikák felhasználásával levezethetünk tulajdonságokat.

Definiáltam továbbá egy egyszerű algebrát, mely segítségével multiplicitások hatékonyan számíthatóak a következtetési feladatokban.

8. fejezet

Automatizált tételbizonyító

Ebben a fejezetben következtető rendszerként automatizált tételbizonyító alkalmazhatóságát mutatom be. Eszközként a Prover9/Mace4 [16] programot használom, mely egyenlőségjeles elsőrendű logikai állítások feldolgozására képes.

8.1. Eszköz bemutatása

Az eszköznek két felhasználási módja van: Prover9 nevű bizonyító és Mace4 nevű ellenpéldakereső. Mindkét funkció három paramétert vár bemenetként:

1. **Feltételek (angolul: Assumptions):** Logikai állítások sorozata, a következtetés premisszáit, axiómát írja le.
2. **Következmények (angolul: Goals):** Olyan tulajdonságok, melyek helyességét a feltételek teljesülése mellett vizsgálni kívánjuk.
3. **Opciók:** Keresést szabályozó paraméterek. Ide tartozik például, hogy mennyi időt szánunk az adott feladatra.

Az első két paraméter mindkét mód esetén megegyezik.

Prover9 alkalmazás feladata, hogy logikailag belássa a következmények helyességét a feltételek helyességéből. Ennek kimeneteként kaphatunk egy helyességet bizonyító levezetést, egy követelmény beláthatatlanságát jelző „exhausted” üzenetet, vagy a keresés sikertelenséget mutató jelzést.

Mace4 funkció olyan véges ellenpéldát keres, amely teljesíti a feltételeket, de a következményt nem. Ha következményként nem adunk meg állításokat, akkor modellkeresőként alkalmazható. Siker esetén felmutat egy feltételeknek megfelelő modellt. Érdemes a két funkciót egyszerre futtatni, mivel egyik sikere biztosítja a másik sikertelenségét, így lerövidíthető a keresési idő.

8.2. Formák leképezése logikai állításokká

A program bemenetként egyenlőségjeles elsőrendű logikai állításokat vár. Állításainkban használhatunk relációkat és függvényeket, változókat és konstansokat, kvantorokat és a

Név	Matematikai jelölés	Program jelölés	Példa
Változó	e, n	`E`, `N` (Nagy betű)	
Konstans	e_1, n_1	`e1`, `n1` (Kis betű)	
Term egyenlőség	$=$	`=`	`A=B`
Term egyenlőtlenség	$\neq$	`!=`	`A!=B`
Negáció	$\neg$	`-`	`-g_Edge(X)`
Diszjunkció	$\vee$	`|`	`a(X)|b(X)`
Konjukció	$\wedge$	`&`	`a(X)&b(X))`
Implikáció	$\rightarrow$	`->`	`a(X)->b(X)`
Univerzális k.	$\forall$	`all`	`all X a(X)`
Egzisztenciális k.	$\exists$	`exists`	`exists X a(X)`

8.1. táblázat. *Logikai jelölések*

megszokott elsőrendű logikai összekötőket. Az alkalmazott nyelvi elemek jelöléseire áttekintést nyújt a 8.1. táblázat.

8.2.1. Új jelölések

Az olvashatóság végett vezessünk be pár új jelölést.

Legyenek $F_1, \ldots F_n$ logikai formulák. Ekkor $\bigoplus\{F_1, \ldots F_n\}$ jelölje azt, hogy az $F_1, \ldots F_n$ formulák közül pontosan 1 igaz. A jelölés pontos meghatározását a függelék 37. definíciója tartalmazza. Látható, hogy a jelölés nem vezet ki az elsőrendű logikából.

Szeretnénk továbbá, ha egy kifejezés több különböző változóra is érvényes legyen. Ennek érdekében vezessünk be két újabb jelölést: $\exists^{\geq m} X$ és $\exists^{=m} X$. $\exists^{\geq m} X$ jelentse azt, hogy van legalább m különböző elem, melyeket behelyettesítve az állítás X változójába az állítás teljesül. $\exists^{=m} X$ pedig jelentes azt, hogy pontosan m különböző elem van. Nevezzük ezt a fajta jelölést multiplicitás kvantornak, melynek leírását a függelék 38. definíciója adja meg.

8.2.2. Gráfstruktúra

Elsőnek fejezzük ki a gráf szerkezetéből adódó logikai állításokat. Logikai állításaink termjei csúcsok és élek lehetnek. Jelöljük a csúcsokat a `g_Node`, az éleket a `g_Edge` predikátumokkal. Ebből származik első állításunk, miszerint egy term az vagy csúcs, vagy él:

```
all X ((g_Node(X) & -g_Edge(X)) | (-g_Node(X) & g_Edge(X))).
```

Egy élnek lehet forrása és célja, jelöljük ezeket a `getSource` és `getTarget` függvényekkel. Tudjuk, hogy a függvények eredménye mindig csúcs:

```
all E (g_Node(getSource(E)) & g_Node(getTarget(E))).
```

Legyen $Lab^N = \{\texttt{lab}_1^N \ldots \texttt{lab}_n^N\}$ a csúcscímkék, $Lab^E = \{\texttt{lab}_1^E \ldots \texttt{lab}_e^E\}$ az élcímkék halmazai. Minden csúcscímkéhez és élcímkéket feleltessünk meg egy predikátumot, melyet

`lab_N_` és `lab_E_` prefixszel jelölünk. Azt, hogy minden csúcsnak és élnek pontosan egy címkéje van, a következőképpen jelölhetjük:

$$\texttt{all N (g\_Node(N)-> } \bigoplus\{\texttt{lab\_N\_lab}_1^N\texttt{(N)},\ldots,\texttt{lab\_N\_lab}_n^N\texttt{(N)}\}\texttt{))}.$$

$$\texttt{all E (g\_Edge(E)-> } \bigoplus\{\texttt{lab\_E\_lab}_1^E\texttt{(E)},\ldots,\texttt{lab\_E\_lab}_e^E\texttt{(E)}\}\texttt{))}.$$

---

22. Példa Legyen $Lab^N = \{\texttt{List},\texttt{Cell},\texttt{Object}\}$ a csúcscímkehalmaz. Ekkor a csúcsok címkézését definiáló állítás a következő:

```
all N (g_Node(N)-> (
(lab_N_Cell(N) & -lab_N_List(N) & -lab_N_Object(N))|
(-lab_N_Cell(N) & lab_N_List(N) & -lab_N_Object(N))|
(-lab_N_Cell(N) & -lab_N_List(N) & lab_N_Object(N)))).
```

Ha gráfjainkhoz tartoznak jólformáltsági kritériumok, akkor ezek is ehhez a szakaszhoz adódnak.

8.2.3. Szürjektív morfizmusból eredő kényszerek

Jelöljük formánk $N = \{\texttt{n}_1 \ldots \texttt{n}_n\}$ csúcsait `s_N_`, $E = \{\texttt{e}_1 \ldots \texttt{e}_e\}$ éleit `s_E_` prefixű predikátumokkal. Azt, hogy egy gráfelem mely formabeli gráfelemhez rendelődik formázás során, ezekkel a predikátumokkal kívánjuk jelölni: ha egy gráfelem a predikátumnak megfelelő formabeli gráfelemhez rendelődött, akkor értéke legyen igaz, különben meg hamis.

Formázás során egy szürjektív morfizmussal rendeljük elemeinket formánkhoz, így predikátumainknak pár feltételt be kell tartaniuk:

1. A morfizmus olyan függvény ami csúcsokat csúcsokhoz, éleket élekhez rendel. Ezért az alábbi állításoknak minden $\texttt{n}_i \in N$ csúcsra és $\texttt{e}_i \in E$ élre teljesülniük kell:

   $$\texttt{all X (s\_N\_n}_i\texttt{(X) -> g\_Node(X)).}$$

   $$\texttt{all X (s\_E\_e}_i\texttt{(X) -> g\_Edge(X)).}$$

2. Minden elemet pontosan egy absztrakt elemhez rendel. Így az alábbi állításoknak teljesülniük kell:

   $$\texttt{all X (g\_Node(X) -> (}\bigoplus\{\texttt{s\_N\_n}_1\texttt{(X)},\ldots,\texttt{s\_N\_n}_n\texttt{(X)}\}\texttt{)).}$$

   $$\texttt{all X (g\_Edge(X) -> (}\bigoplus\{\texttt{s\_E\_e}_1\texttt{(X)},\ldots,\texttt{s\_E\_e}_e\texttt{(X)}\}\texttt{)).}$$

3. A szürjektivitása miatt minden absztrakt elemhez kell tartozzon konkrét. Azaz minden $\texttt{n}_i \in N$ csúcsra és $\texttt{e}_i \in E$ élre:

   $$\texttt{exists X s\_N\_n}_i\texttt{(X).}$$

`exists X s_E_e`$_i$`(X).`

4. A morfizmus miatt a forma megköti, hogy egy konkrét él mely konkrét csúcsokkal lehet kapcsolatban. Ha egy él egy $\mathtt{e}_i$ absztrakt csúcsba képződött, akkor annak forrása $src(\mathtt{e}_i) = \mathtt{e}_i^{src}$-be, célja $trg(\mathtt{e}_i) = \mathtt{e}_i^{trg}$-be kerül. Ennek megfelelően minden élre ki kell jelentenünk:

   `all E (s_E_e`$_i$`(E) -> s_N_n`$_i^{src}$`(getSource(E))).`

   `all E (s_E_e`$_i$`(E) -> s_N_n`$_i^{trg}$`(getTarget(E))).`

A függelék 9.3. példájában megtekinthető egy kidolgozott eset gráfstruktúrákból és szürjektív leképezésekből származó állításokra.

8.2.4. Ekvivalenciák és reprezentációk

Formáink lehetséges konkretizáltjait gráfelemekre címkézett a reprezentációk is megszabják. Ezek újabb axiómákat generálnak.

- **Élcímke (címke: *olab*, reprezentáció: (`lab`)):** Egy forma *olab* szerinti címkézése meghatározza a vele összeegyeztethető élek címkéjét. Így minden `e` élhez tartozik egy címkézést jelző állítás:

  `all E (s_E_e(E) -> lab_E_lab(E)).`

- **Ekvivalencia szorzat (ekvivalencia: $\overset{A}{\sim} \times \overset{B}{\sim}$, reprezentáció: (a,b)):** Ilyen alakú ekvivalencia esetén $\overset{A}{\sim}$-ra a reprezentációval és $\overset{B}{\sim}$-re b reprezentációval teljesülő állításokat kell megfogalmazni. Ez visszavezethetőek két másik axióma-generálás feladatra, melyek eredményeit egyszerűen összefűzzük.

- **Csúcsokon értelmezet predikátum (ekvivalencia: $\overset{Pred}{\underset{P}{\sim}}$, reprezentáció: v):** Legfőbb feladatunk átírni a P csúcspredikátum Prover9-nak megfelelő bemenetre alakítása. Ez legtöbb esetben magától értetődő feladat (mint például $isList(n)$, $firstCell(n)$) de például tranzitív lezártat tartalmazó formulánál (mint például a $reachable(n)$) nehézségekbe ütközünk, mivel kilép ez elsőrendű logika kereteiből. Ha leképezhető a predikátum, akkor az `n` csúcsra v értékének megfelelően a következő állítás tejlesül:

  - **0 esetén:** `all X (s_N_n(X) -> -`P`(X)).`
  - **1 esetén:** `all X (s_N_n(X) ->  `P`(X)).`

- **Csúcshalmazon értelmezet predikátum (ekvivalencia: $\overset{IPred}{\underset{P}{\sim}}$, reprezentáció: v):** Feladatunk ugyanaz, mint a csúcson értelmezett predikumok esetén. Ha $v = 1/2$ akkor annyit tudunk a predikátumról, hogy néha teljesül, néha nem. Tehát `n` csúcsra v értékének megfelelően a következő állítás tejlesül:

– **0 és 1 esetén:** ugyanaz, mint az előző pontban.

– $1/2$ **esetén:**

$$\texttt{exists X exists Y (X!=Y \& s\_E\_n(X) \& s\_E\_n(Y) \&} \; P(\texttt{X}) \; \texttt{\& -}P(\texttt{Y})).$$

- **Bizonytalan él (ekvivalencia: $\overset{Alw}{\sim}$, reprezentáció: v):** Ha v értéke igaz, akkor a `source` és `target` csúcsok között futó `e` élre teljesül, hogy minden `source`-beli konkrét csúcsból vezet `e`-beli él minden `target`-beli csúcsba. Ezt az alábbi módon tudjuk formalizálni v értékének függvényében:

  – **1 esetén:**

$$\texttt{all S all T exists E ((s\_N\_source(S) \& s\_N\_terget(T)) ->}$$
$$\texttt{(s\_N\_source(getSource(E)) \& s\_N\_target(getTarget(E)) \& s\_E\_e(E))).}$$

  – $1/2$ **esetén:** a $v = 1$ eset tagadása.

- **Multiplicitások (ekvivalencia: $\underset{m}{\overset{NM}{\sim}}$ és $\underset{m}{\overset{EM}{\sim}}$, reprezentáció: i):** Esetek számosságának jelölésére bevezettük a multiplicitás kvantorokat. Így egyszerűen definiálhatóak a multiplicitásból származó axiómák. Az `n` formabeli csúcsra az alábbi állítás igaz:

  – $i \neq \omega$ **esetén:** $\exists^{=i}$`X s_N_n(X).`

  – ω **esetén:** $\exists^{\geq m+1}$`X s_N_n(X).`

  Az élek multiplicitása is hasonlóképp definiálható.

- **Szomszédsági ekvivalencia (ekvivalencia: $\underset{r,p}{\overset{Neigh}{\sim}}$, reprezentáció alakja: `l` vagy $(nrep_{r-1,p}, IN, OUT)$):** Ha reprezentációnk egy egyszerű `l` élcímkéből áll, akkor `n` csúcsunkról az alábbi állítást tudjuk tenni:

$$\texttt{all N (s\_N\_n(N) -> lab\_N\_l(N)).}$$

  Ha $(nrep_{r-1,p}, IN, OUT)$ alakú reprezentációnk van, háromféle állításhalmaz teljesül csúcsunkra:

  1. $nrep_{r-1,p}$-ből kapunk egy eggyel kisebb sugarú reprezentációt. Ebből rekurzívan generáljuk le egy egyel kisebb sugarú szomszédság segítségével a szükséges állításokat.
  2. IN-be olyan bejegyzések vannak, melyek leírják: hány és milyen kimenő él van csúcsunkból, és milyen szomszédságú csúcsba vezet. Ha van $m \neq \omega$ darab `l` címkéjű kimenő él, amely `n` csúcsunkból P_{neigh} predikátummal leírt csúcsba vezet, az alábbi állítást kapjuk:

$$\texttt{all N } \exists^{=m}\texttt{E (s\_N\_n(N) ->}$$
$$\texttt{(lab\_E\_l(E) \& N=getSource(E) \&} \; P_{neigh}(\texttt{getTarget(E)}))).$$

$m = \omega$ esetén állításunkban $\exists^{=m}$ helyett $\exists^{\geq p+1}$-et használunk, így jelezvén a szomszédság pontatlan ismeretét.

3. Kimenő élekről bejegyzéseket tároló OUT is hasonlóképpen állítható elő, itt a `getSource` és a `getTarget` függvények felcserélődnek.

Miután minden gráfelem minden reprezentációjából előállítottuk az állításainkat, a gráf struktúrájából és a szürjektív morfizmusból előálló axiómák után megkapjuk formánk logikai ábrázolását.

8.2.5. Fókuszált formák

Fókuszált formákat is szeretnénk kezelni tételbizonyítónkkal. A konkrét gráfelemeket konstansokkal ábrázoljuk, melyeket csúcsok esetén `g_N_`, élek esetén `g_E_` prefix jelöl.

Konkrét gráfelemeinkre ki kell kötnünk, hogy egyikük sem lehet egyenlő a másikkal. Ha fókuszált formákkal dolgozunk, enyhítenünk kell a szürjektív morfizmusból adódó megkötéseket. A 2. pont a következőképp változik, ha fókuszált csúcsainkat `g_N_n1` ... `g_N_nn`, éleinket `g_E_e1` ... `g_E_ee` jelöli:

$$\texttt{all X (g\_Node(X) -> (}\bigoplus\{\texttt{s\_N\_n}_1\texttt{(X)},\ldots,\texttt{s\_N\_n}_n\texttt{(X)},\texttt{X=g\_N\_n1},\ldots,\texttt{X=g\_N\_nn}\}\texttt{)).}$$

$$\texttt{all X (g\_Edge(X) -> (}\bigoplus\{\texttt{s\_E\_e}_1\texttt{(X)},\ldots,\texttt{s\_E\_e}_e\texttt{(X)},\texttt{X=g\_E\_e1},\ldots,\texttt{X=g\_E\_ee}\}\texttt{)).}$$

Konkrét elemek esetén gráfunk szerkezetét megadhatjuk a `getSource` és a `getTarget` függvények konstans értékekre történő definiálásával. Így például ha a `g_E_e`-vel jelölt él célja a `g_N_n`-nel jelölt konkrét csúcs, azt így fejezzük ki:

```
getTarget(g_E_e)=g_N_n.
```

Abban az esetben, ha az `s_N_n` absztrakt csúcsba vezet az él, így írhatjuk le:

```
s_N_n(getTarget(g_E_e)).
```

8.3. Lekérdezések

Ebben a szakaszban bemutatom az absztrakt transzformátor következtetési feladatainak tételbizonyítóval történő megvalósítását. Minden esetben onnan indulunk ki, hogy konkrét lehetőségeink vannak, és azokat vizsgáljuk. Mivel minden lépésben a lehetőségek száma véges, így előszűrés nélkül is megvalósítható az ilyen vizsgálat.

8.3.1. Konkrét illesztés

Írjuk át illesztendő gráfunk csúcsait `N1` ... `Nn`, éleit `E1` ... `Ee` egzisztenciálisan kvantált változókká. A megfelelő címkézés szerint teljesülnek a gráfelemekre a címkéket jelölő predikátumok, valamint a `getSource` és a `getTarget` függvények is a megfelelő gráfcsúcsot rendelik éleinkhez.

Egy illeszkedés a következő esetben helyes: felírjuk minden változóra, hogy teljesül rá az illesztett absztrak gráfelem predikátuma. Illeszkedéseinket úgy vizsgáljuk meg, hogy feltételekként felírjuk formánk logikai ábrázolását, következményként a helyes illeszkedés tagadását. Ha nem tudjuk belátni, hogy nem illeszkedhet formánkra az adott módon a minta, a lehetőséget megtartjuk, különben elvetjük.

8.3.2. Illeszkedések felsorolása

Egy konkrét absztrakt gráfekem egy instrumentális reprezentációjának lehetőségét vizsgáljuk. A reprezentáció lehetőségét átírjuk S logikai állításokká, majd megpróbáljuk belátni: ha minden, fókuszáláskor lértrejövő konkrét gráfelemre teljesültek az absztrakt gráfelemre szóló állítások, akkor az S nem teljesülhet.

Ennek érdekében a tételbizonyítónak feltételekként megadjuk formánk logikai ábrázolását, a fókuszáláskor létrejövő elemek konstansait és a velük kapcsolatos strukturális és címkézési ismereteket. Következményként megpróbáljuk belátni, hogy S nem teljesülhet azokra a gráfelemekre, melyek nem konkrétak, de és részei az absztrakt gráfelemnek. Ha sikerül ezt belátni, a lehetőséget elvethetjük.

8.3.3. Reprezentációk vizsgálata

Ebben a feladatban már ténylegesen fókuszált formával dolgozunk. Ellenőrizni szeretnénk, hogy van-e olyan reprezentáció formánkban, ami nem teljesülhet.

Ennek érdekében formánk majdnem teljes logikai ábrázolását feltételként adjuk meg, mindig egy reprezentációhoz tartozó állításhalmazt hagyunk ki. Ennek a az állításhalmaznak a tagadását próbáljuk meg bizonyítani, és ha sikerül, biztos ellentmondás van fókuszált formánkban. Így az ilyen lehetőségeket eldobjuk.

8.3.4. Reprezentációk átírása

Feladatunk egy konkrét lehetséges reprezentáció vizsgálata. Azt kívánjuk megvizsgálni, hogy ha fókuszált formánkban átírás előtt állításaiból, akkor az átírás eredményeként megkapott új fókuszált formának állításaiból következtetve megengedhetjük-e a lehetséges új reprezentációt.

A következtetés végrehajtásához létrehozunk új `new_g_Node` és `new_g_Edge` predikátumokat és `new_getSource` és a `new_getTarget` függvényeket. Ezekkel megadjuk, hogy mely elemek maradnak meg, melyek változnak, úgy hogy az új gráfelemekre az eredeti állítások nem igazak, az eltűnőkre pedig az újak. Absztrakt gráfelemek esetén minden példány változatlan marad, azaz ha eg `X` változó nem egyenlő egyik konstanssal sem akkor `new_g_Node(X) <->g_Node(X)`. Ilyen feltételekkel próbáljuk belátni, hogy egy reprezentációnak megfelelő állítások, melyekben már az új predikátumok és függvények szerepelnek, nem teljesülhetnek. Ha sikerül a bizonyítás, az adott reprezenáció lekerül a lehetőségek listájáról.

8.3.5. Reprezentációk számítása

Ismét fókuszált formán dolgozunk. Feladatunk két egyszerűen megfogalmazható kategóriába tartozhat:

1. Lehet-e egy konkrét elemnek egy adott reprezentációja?
2. Lehet-e egy elemcsomportnak egy adott reprezentációja?

Mindkét esetben feltételnek a fókuszált forma logikai ábrázolását adjuk meg, majd megpróbáljuk belátni az reprezentációhoz tartozó állítások ellenkezőjét. Sikeres esetben elvethetjük a lehetőséget.

8.3.6. Konzisztenciavizsgálat

Egy formában ellentmondásosságának felfedésére két lehetőségünk van:

- **Ellentmondás levezetése:** Formánk logikai ábázolásából meg próbáljuk levezetni az üres állítást. Ha ez sikerül, formánk tartalmaz ellentmondást.
- **Modell keresése:** Megpróbálunk ellenpéldát keresni a forma feltételeire, és üres következményre. Ha találunk ilyet formánk biztosan nem ellentmondásos.

8.3.7. Biztonság ellenőrzése

A biztonsági kritérium egy gráfra megfogalmazott feltétel. Ha ezt a feltételt fel tudjuk írni elsőrendű logikai állításokként, következtetéseket végezhetünk teljesülésének vézsgálata céljából. Ha formánk logikai állításaiból megpróbáljuk levezetni a kritérium tagadását. Ha sikerül, formánk biztonságos.

8.4. Értékelés

Automatizált tételbizonyító segítségével tranzitív lezárt kivételével formánk teljes szemantikáját leképezhetjük következtetési terünkbe. Ennek köszönhetően ez a módszer adja a legpontosabb eredményeket következtetési feladatokra. Másik oldalról viszont ez a következtető rendszer rendelkezik a legnagyobb számításigénnyel, tovább nehezít, hogy minden lehetőséget egyesével kell ellenőriznie. Ezért mindenféleképpen érdemes a következtetési feladatok lehetőségeit előszűrni.

A [11] cikkben leírt eljárás háromértékű logikában dolgozó következtető rendszer számításaira épít. Legnagyobb különbségként az tüntethető fel, hogy végig egy- és kétparaméteres relációkkal dolgozik dolgozik a módszer, ahol a relációk teljesülésének 0, 1 vagy $1/2$ értékét őrzik az állapotok. Esetemben az a következtetés egyik lépésében sem távolodok el gráfstruktúrától.

Tapasztalataim szerint a Prover9 alkalmasnak bizonyul következtetési feladatok precíz számítására, és a későbbiekben teljesítménye is javítható lesz a keresési opciók (mint például az állítás sorrendezés) hatékony konfigurációja mellett. A Mace4 komponens azonban már 10-12 gráfelemet tartalmazó formák konkretizáltjainak keresésénél is gyengének bizonyul, sikere véletlenszerűnek tűnik.

8.5. Összefoglalás

Ebben a fejezetben megadtam egy általam kifejlesztett eljárást, hogyan lehet formáink tulajdonságait egy elsőrendű logikai tételbizonyítóval következtetni. Leírtam egy leképezést, mely formáinkat a Prover9/Mace4 nevű külső eszköz bemenetének megfelelő logikai állításokká alakítja. A dolgozat során bemutatott absztrakciókra is levezettem az állításokat. Ezek után megadtam, hogyan lehet az absztrakt transzformáció következtetési feladatait megvalósítani bizonyításkeresési feladatokkal.

9. fejezet

Konklúzió

9.1. Eredményeim

A dolgozatom legfőbb saját elméleti tudományos eredményei:

- Megadtam a formák és a formatranszformációk egységes és általánosított definícióját, amely lefedi a szakirodalomban ismert legnépszerűbb forma-definíciókat.
- Kidolgoztam egy általános és bővíthető keretrendszert a formatranszformációkon alapuló verifikáció támogatására, amely lehetővé teszi mind a formázást definiáló absztrakció, mind pedig az analízis során felhasznált következtető rendszerek kombinált felhasználását.
- Leképezést definiáltam a formákról példányszintű és nyelvszintű következezetetést támogató rendszerekbe, amely lehetővé teszi a formázás során felmerülő következtetési feladatok elvégzését.

Dolgozatom a "Modelltranszformációk formális analízise" című BSc szakdolgozatomon alapul, bár a fenti eredmények közül mindössze az általánosított forma definíció volt közölve, az egységes ellenőrző architektúra és a leképezések definiálása a TDK munkám részét képzezik.

Dolgozatomhoz kapcsolódó főbb gyakorlati, implementációs eredményeim a következők:

- Formák ábrázolása és tárolása és kezelése a VIATRA2 rendszerben.
- Hatékony állapot-ekvivalencia vizsgálat megoldása.
- A bemutatott absztrakciókkal történő formázás megvalósítása.
- Formák vizualizálása yEd [18] eszközzel.

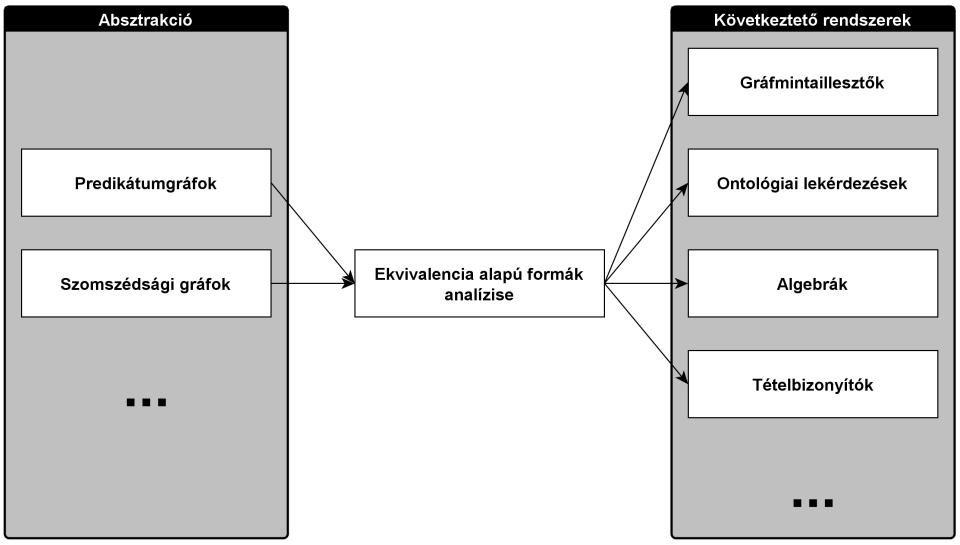
9.2. Értékelés

Dolgozatomban megadtam egy új verifikációs módszert, amely végtelen állapotterű rendszerek helyességellenőrzését végezi. A módszer a szakirodalomban megtalálható gráfabsztrakcióval kapcsolatos ismeretanyagok főbb eredményein alapszik, és ezen meglévő tudás integrált alkalmazását és kiterjesztését célozza.

Absztrakt állapottér építése során a formák precizitásának növelése és az állapottér méretének csökkentése a cél. Tipikus esetben egyik eredmény javítása a másik romlását vonja maga után. Ezen az összefüggésen **problémaspecifikus absztrakcióval** léphetünk túl, mivel így pontosabb következtetéseket vonhatunk formáinkból, úgy, hogy ezzel nem szaporítjuk a helyességellenőrzés szempontjából érdektelen részleteket.

Az eddigi eljárások törékenyeknek bizonyultak **bővíthetőség** szempontjából, ezért a **kiterjeszthetőség** érdekében szétcsatoltam az alkalmazott absztrakciókat a felhasznált következtető rendszerektől (lásd 9.1. ábra).

Az így előállt keretrendszerhez ekvivalenciarelációk formájában újabb absztrakciók adhatóak, melyekkel egyre több lehetőség nyílik a **feladatspecifikus paraméterezésre**. Egy újabb ekvivalencia megadása nincs hatással az egész architektúrára, így ilyenek akár egyedi feladatra is megadhatóak. A következtetési lépésekhez is újabb megoldások csatolhatóak, minden újabb módszer javítja az absztrakció precizitását, és ügyes feladatelosztás esetén feladatok szűrésével még a következtetés műveletigénye is csökkenthető.



9.1. ábra. *Absztrakció és következtető rendszer szétcsatolása.*

Az így kapott rendszer alapvetően teljesítménnyel fizet pontosságáért és bővíthetőségéért. Egy állapot mérete többszörösen meghaladhatja a hasonló feladatokra készített egyedi megoldásokét, hiszen azok kifejlesztésénél az volt az egyik elsődleges cél, hogy minél több állapot lehessen tárolni. Másrészről a következtetések is lassabbak, mint a konkrét feladatokra optimalizált eljárásoknál.

Ugyanakkor a kombinált megoldás segítségével olyan végtelen állapotterű verifikációs feladatok megoldása válik lehetségessé, amelyet az egyedi absztrakciók nem támogatnak.

9.3. Fejlesztési lehetőségek

A keretrendszer paraméterezhetősége céljából újabb absztrakciók keresendőek. Ennek eléréséhez komplex esettanulmányok vizsgálata szükséges, melyek tapasztalataiból tanulva rálelhetünk azokra a tulajdonságokra, melyek tipikus felhasználás esetén jól osztályozzák

gráfelemeinket.

A számítások precizitásának és hatékonyságának növelése érdekében új következtető rendszerek bevezetése, és a meglévő rendszerek konfigurációjának, alkalmazási módjának javítása a cél. Kitüntetett figyelmet érdemelnek ilyen téren a *leíró logikán következtető eljárások*. Ezek vizsgálata közben a különböző nyelvcsaládok kifejezőerejét és következtetési hatékonyságát is fel kell mérni.

Jelenlegi állapotában rendszerünk egyetlen formán végez csak következtetéseket. Érdekes és eredményekkel kecsegtető terület annak vizsgálata, hogy hogyan építhetőek bele egyes állapotterekben megismert eredmények más állapottér számításába. Egy kevésbé precíz állapottér heurisztikákkal és sikeresen bejárt állapotokkal javíthatja és gyorsíthatja számításainkat.

Eddigi munkánkban gráftranszformációinkat tetszőlegesen alkalmazhattuk minden állapotra. Gráfnyelvtanunk ellenőrzési technikája kibővíthetőnek tűnik modelltranszformációs rendszer ellenőrzésére is, mellyel bonyolultabb rendszereket is ellenőrizhetünk.

Függelék

A. Gráfok, gráftranszformációk

11. Definíció (Morfizmus). *Legyenek $G, H \in \mathcal{G}$. Ekkor a G és H gráfok közötti morfizmuson azt az $m : N_G \cup E_G \mapsto N_H \cup E_H$ függvényt értjük, amelyre teljesülnek az alábbi feltételek:*

- *m csúcsokhoz csúcsokat, élekhez éleket rendel:*

$$\forall n(n \in N_G \Rightarrow m(n) \in N_H) \wedge \forall e(e \in E_G \Rightarrow m(e) \in E_H)$$

- *ha G-ben egy n csúcs egy e élnek forrása vagy célja volt, ez maradjon igaz $m(n)$-re és $m(e)$-re is:*

$$\forall n \in N_G, \forall e \in E_G \Big(\Big(n = src_G(e) \Rightarrow m(n) = src_H(m(e)) \Big) \wedge \Big(n = trg_G(e) \Rightarrow m(n) = trg_H(m(e)) \Big) \Big)$$

Az m-et címkehelyes morfizmusnak nevezzük, ha ezeken kívül az alábbi feltételek is teljesülnek:

- *G és H címkéi közösek: $Lab_G^N = Lab_H^N \wedge Lab_G^E = Lab_H^E$*
- *A csúcsok és élek címkéi ugyanazok maradnak:*

$$\forall n \in N_G \Big(lab_G^N(n) = lab_H^N(m(n)) \Big) \wedge \forall e \in E_G \Big(lab_G^E(e) = lab_H^E(m(e)) \Big)$$

Az m morfizmus esetén m^N jelentse csak a csúcsokon értelmezett függvényt, m^E pedig csak az éleken értelmezettet.

12. Definíció (Injektív, szürjektív, bijektív függvény). *Legyen $f : A \mapsto B$ függvény. Azt mondjuk, hogy az f függvény* injektív, *ha minden $x, y \in A$-ra teljesül, hogy $f(x) = f(y)$ esetén $x = y$. Az f függvény* szürjektív, *ha minden $b \in B$-re teljesül, hogy létezik olyan $x \in A$ amelyre $f(x) = b$. Az f függvény* bijektív, *ha injektív és szürjektív egyszerre.*

13. Definíció (Gráftranszformáció). *Egy $Prod = (LHS, RHS)$ gráftranszformációs szabály esetén különböztessük meg az alábbi, $Prod$-dal indexelt halmazokat:*

- *A törlődő csúcsokat* $N^{del}_{Prod} = N_{LHS} \setminus N_{RHS}$*, a törlődő éleket* $E^{del}_{Prod} = E_{LHS} \setminus E_{RHS}$ *jelöli.*

- *Az új csúcsokat* $N^{new}_{Prod} = N_{RHS} \setminus N_{LHS}$*, az új éleket* $E^{new}_{Prod} = E_{RHS} \setminus E_{LHS}$ *jelöli.*

Legyen G egy címkézett gráf, $Prod = (LHS, RHS)$ egy gráftranszformációs szabály. G elemei diszjunktak L és RHS elemeitől, ellenkező esetben vegyünk L és RHS helyett egy velük izomorf diszjunkt gráfot, és azokkal dolgozzunk tovább. Legyen $Prod \xrightarrow{m} G$ amire igaz, hogy $Prod$ nem törlődő éle nem illeszkedik törlődő csúcshoz:

$$\forall e \in E_G \Big(src_G(e) \in m(N^{del}_{Prod}) \vee trg_G(e) \in m(N^{del}_{Prod}) \Rightarrow e \in m(E^{del}_{Prod}) \Big)$$

A $Prod$ transzformációs szabályt m illeszkedéssel G-n alkalmazva a H címkézett gráfot kapjuk, és ezt $G \overset{Prod,m}{\longrightarrow} H$-val jelöljük, ha

$$\begin{aligned}
N_H &= (N_G \setminus m(N^{del}_{Prod})) \cup N^{new}_{Prod} \\
Lab^N_H &= Lab^N_G \\
lab^N_H(n) &= \begin{cases} lab^N_G(n) & ha\ n \in N_G \setminus m(N^{del}_{Prod}) \\ lab^N_{RHS}(n) & ha\ n \in N^{new}_{Prod} \end{cases} \\
E_H &= (E_G \setminus m(E^{del}_{Prod})) \cup E^{new}_{Prod} \\
src_H(e) &= \begin{cases} src_H(e) & ha\ e \in E_G \setminus m(E^{del}_{Prod}) \\ src_{RHS}(e) & ha\ e \in E^{new}_{Prod} \wedge src_{RHS}(e) \in N^{new}_{Prod} \\ m(src_{RHS}(e)) & ha\ e \in E^{new}_{Prod} \wedge src_{RHS}(e) \notin N^{new}_{Prod} \end{cases} \\
trg_H(e) &= \begin{cases} trg_H(e) & ha\ e \in E_G \setminus m(E^{del}_{Prod}) \\ trg_{RHS}(e) & ha\ e \in E^{new}_{Prod} \wedge trg_{RHS}(e) \in N^{new}_{Prod} \\ m(trg_{RHS}(e)) & ha\ e \in E^{new}_{Prod} \wedge src_R(e) \notin N^{new}_{Prod} \end{cases} \\
Lab^E_H &= Lab^E_G \\
lab^E_H(e) &= \begin{cases} lab^E_G(e) & ha\ e \in E_G \setminus m(E^{del}_{Prod}) \\ lab^E_R(e) & ha\ e \in E^{new}_{Prod} \end{cases}
\end{aligned}$$

14. Definíció (Elérhető állapot). *Legyen $G_0, H \in \mathcal{G}(Lab^N, Lab^E)$ címkézett gráf, R gráftranszformációs szabályok, P pedig jólformáltsági kritériumok halmaza. Ekkor G_0-ból R-beli transzformációkkal elérhető H (jelölésben: $G_0 \xrightarrow{R,P} H$), ha léteznek olyan $Prod_0, \ldots, Pprod_n \in R$ transzformációk, $m_0, \ldots m_n$ illeszkedések és $G_1, \ldots G_n \in \mathcal{G}(Lab^N, Lab^E)$ címkézett gráfok, hogy:*

$$\forall G \in \{G_0 \ldots, G_n, H\}, \forall P_i \in P (G \models P_i)$$

$$G_0 \overset{Prod_0,m_0}{\longrightarrow} G_1, G_1 \overset{Prod_1,m_1}{\longrightarrow} G_2, \ldots, G_n \overset{Prod_n,m_n}{\longrightarrow} H$$

Egy gráf önmagából definíció szerint elérhető.

15. Definíció (Állapottér). *Legyen $G_0 \in \mathcal{G}(Lab^N, Lab^E)$ címkézett kiinduló gráf, R pedig gráftranszformációs szabályok, P pedig jólformáltsági kritériumok halmaza. Ekkor a $states(G_0, R, P)$ függvény eredménye egy olyan $STATES \in \mathcal{G}(\mathcal{G}(Lab^N, Lab^E), R)$ gráfokkal csúcscímkézett és transzformációkkal élcímkézett gráf, amelyre az alábbi állítások igazak:*

- *Minden csúcscímke csak egyszer szerepel, más szóval lab^N_{STATES} injektív.*

- *Csúcsai azon G gráfokkal vannak címkézve, amelyekre teljesül, hogy $G_0 \overset{R,P}{\longrightarrow} G$.*
- *Minden olyan G és H gráfhoz, m morfizmushoz és $Prod \in R$ transzformációhoz, amelyre teljesülnek, hogy $G \overset{R,G_0}{\longrightarrow}$, $H \overset{R,G_0}{\longrightarrow}$ és $G \overset{Prod,m}{\longrightarrow} H$ pontosan egy él tartozik. Ennek az élnek a címkéje $Prod$, irányát tekintve a G-vel címkézett csúcsból vezet a H-val címkézett csúcsba.*

B. Ekvivalenciarelációk, Formák, formázás

16. Definíció (Ekvivalenciareláció). *Egy H halmaz fölött értelmezett $\sim \subseteq H \times H$ reláció* ekvivalenciareláció, *ha $\sim$-ra az alábbi feltételek teljesülnek:*

1. *reflexív, azaz minden $a \in H$ esetén $a \sim a$*
2. *szimmetrikus, azaz minden $a, b \in H$ esetén $a \sim b \Rightarrow b \sim a$*
3. *tranzitív, azaz minden $a, b, c \in H$ esetén $a \sim b \wedge b \sim c \Rightarrow a \sim c$*

17. Definíció (Ekvivalenciaosztály). *Egy H halmaz fölött értelmezett egy $\sim \subseteq H \times H$ ekvivalenciareláció. Ekkor egy $a \in H$ elemhez tartozó ekvivalenciaosztály a következő:*

$$[a]_\sim = \{x | a \sim x\}$$

Ha $K \subseteq H$, a K halmaz elemeit tartalmazó ekvivalenciaosztályok pedig a kövezkezők:

$$K/\sim = \{[x]_\sim | x \in K\}$$

18. Definíció (Reprezentáló függvény). *Legyen H egy halmaz, és legyen a H halmazon értelmezett $\sim \subseteq H \times H$ ekvivalencia. A $rep : H \mapsto C$ függvényt* reprezentáló függvénynek, *C nevezzük elemeit* reprezentációknak, *ha az alábbi állítás teljesül*[1]*:*

$$\forall a, b \in H \Big(a \sim b \Leftrightarrow rep(a) = rep(b) \Big)$$

19. Definíció (Ekvivalencia szorzat). *Legyen H egy halmaz, és legyenek $\overset{E}{\sim}, \overset{F}{\sim} \subseteq H \times H$ ekvivalenciarelációk. Ekkor $(\overset{E}{\sim} \times \overset{F}{\sim}) \subseteq H \times H$ is egy ekvivalenciareláció, amely a következőképen van definiálva:*

$$\forall a, b \in H \Big(a \overset{E}{\sim} \times \overset{F}{\sim} b \Leftrightarrow a \overset{E}{\sim} b \wedge a \overset{F}{\sim} b \Big)$$

Továbbá ennek nagyoperátoros megfelelője a következő:

$$\prod_{i=1}^{n} \overset{E_i}{\sim} = \overset{E_1}{\sim} \times \ldots \times \overset{E_n}{\sim}$$

2. Állítás (Ekvivalencia szorzatot reprezentáló függvény). *Legyen H egy halmaz, legyenek $\overset{E_1}{\sim}, \ldots \overset{E_n}{\sim} \subseteq H \times H$ olyan ekvivalenciarelációk, amelyeket az $f_1 : H \mapsto C_1, \ldots, f_n : H \mapsto C_n$ függvények reprezentálnak. Ha $F = \prod_{i=1}^{n} \overset{E_i}{\sim}$, akkor azt reprezentálja az alábbi $f_F : H \mapsto C_1 \times \ldots \times C_n$ függvény:*

$$f_F(h) = \Big(f_1(h), \ldots, f_n(h) \Big)$$

[1] A C elemei fölött értelmezett $=$ ekvivalenciának nem feltétlenül a logikai egyenlőséget kell jelölnie, lehet tetszőleges, reprezentációk fölött értelmezett ekvivalencia.

3. Állítás (Formák számossága). *Legyenek* $Comp$, $Instr^N$, Lab^E *és* $Instr^E$ *véges halmazok. Ekkor* $|\mathcal{S}(Comp, Instr^N, Lab^E, Instr^E)|$ *is véges. (A bizonyítás az alábbi cikkekben tallható: [12, 13].)*

20. Definíció (Általánositott inverz függvény). *Egy* $f : A \mapsto B$ *függvény* általánosított inverzén *azt az* $\text{inv}\, f : B \mapsto 2^A$ *függvényt értjük, amire:*

$$\forall b \in B \Big(\text{inv}\, f(b) = \{a | f(a) = b\} \Big)$$

Többváltozós $f : A, A_1, \ldots, A_n \mapsto B$ *függvény esetén* $\text{inv}\, f : B, A_1, \ldots, A_n \mapsto 2^A$, *ahol:*

$$\forall b \in B, \forall a_1 \in A_1 \ldots a_n \in A_n \Big(\text{inv}\, f(b, a_1, \ldots, a_n) = \{a | f(a, a_1, \ldots, a_n) = b\} \Big)$$

$\neg$	
0	1
1	0
$1/2$	$1/2$

$\wedge$	0	1	$1/2$
0	0	0	0
1	0	1	$1/2$
$1/2$	0	$1/2$	$1/2$

$\vee$	0	1	$1/2$
0	0	1	$1/2$
1	1	1	1
$1/2$	$1/2$	1	$1/2$

$\Rightarrow$	0	1	$1/2$
0	1	1	1
1	0	1	$1/2$
$1/2$	$1/2$	1	$1/2$

$\cup$	0	1	$1/2$
0	0	$1/2$	$1/2$
1	$1/2$	1	$1/2$
$1/2$	$1/2$	$1/2$	$1/2$

, $\bigcup_{i=1}^{n} = P_1 \cup \ldots \cup P_n$

1. táblázat. *Háromértékű logikán értelmezett operátorok.*

C. Paraméter ekvivalenciák

21. Definíció (Multiplicitás reprezentáció). *Egy adott $m \in \mathbb{N}^+$ esetén M_m legyen az alábbi halmaz:*

$$M_m = \{1, \ldots, m, \omega, undef\}$$

Legyen továbbá $mult_m : \mathbb{N} \mapsto M_m$ az alábbi függvény:

$$mult_m(i) = \begin{cases} undef & ha\ i = 0 \\ i & ha\ 0 < i \leq m \\ \omega & m<i \end{cases}$$

22. Definíció (Csúcsmultiplicitás). *Legyen $\overset{NM}{\underset{m}{\sim}} \subseteq 2^{\mathcal{N}} \times 2^{\mathcal{N}}$, ahol $m \in \mathbb{N}^+$ a következő ekvivalenciareláció:*

$$\forall U, V \in 2^{\mathcal{N}} \Big(U \overset{NM}{\underset{m}{\sim}} V \Leftrightarrow |U| = |V| \ \vee \ (m < |U| \wedge m < |V|) \Big)$$

Ekkor a $\overset{NM}{\underset{m}{\sim}}$ a $mult_m^N : 2^{\mathcal{N}} \mapsto M_m$ függvény által reprezentált ekvivalenciareláció:

$$mult_m^N(U) = mult_m(|U|)$$

23. Definíció (Élmultiplicitás). *Legyen $\overset{NM}{\underset{m}{\sim}} \subseteq 2^{\mathcal{N}} \times 2^{\mathcal{N}}$, ahol $m \in \mathbb{N}^+$ a következő ekvivalenciareláció:*

$$\forall U, V \in 2^{\mathcal{N}} \Big(U \overset{NM}{\underset{m}{\sim}} V \Leftrightarrow |U| = |V| \ \vee \ (m < |U| \wedge m < |V|) \Big)$$

Ekkor a $\overset{NM}{\underset{m}{\sim}}$ a $mult_m^N : 2^{\mathcal{N}} \mapsto M_m$ függvény által reprezentált ekvivalenciareláció:

$$mult_m^N(U) = mult_m(|U|)$$

24. Definíció (Bizonytalansági ekvivalencia). *Legyen $\overset{Alw}{\sim} \subseteq (2^{\mathcal{N}} \times 2^{\mathcal{E}} \times 2^{\mathcal{N}})^2$ ekvivalencia az $alw : (2^{\mathcal{N}} \times 2^{\mathcal{E}} \times 2^{\mathcal{N}})^2 \mapsto \{0, 1, 1/2\}$ által reprezentált függvény, amely a követke-*

zőképpen van definiálva:

$$sum((S,E,T)) = \begin{cases} 1 & ha\ \forall s \in S, \forall t \in T, \exists e \in E(src(e) = s \wedge trg(e) = t) \\ 1/2 & különben \end{cases}$$

25. Definíció (Szomszédsági ekvivalencia). *Egy adott $r \in \mathbb{N}$ és $p \in \mathbb{N}^+$ esetén az r távolságú p párhuzamosságú* szomszédsági ekvivalencián *az alábbi $\overset{Neigh}{\underset{r,p}{\sim}} \subseteq \mathcal{N} \times \mathcal{N}$ ekvivalenciát értjük (ne feledjük, hogy $mult_m(|\emptyset|) = undef$):*

- *$r = 0$ esetén: $u \overset{Neigh}{\underset{0,p}{\sim}} v$ ha ugyanaz a csúcsok címkéje: $lab^N(u) = lab^N(v)$*
- *$r > 0$ esetén: $u \overset{Neigh}{\underset{r,p}{\sim}} v$ ha az alábbi állítások teljesülnek:*
  - *$r-1$ sugárban is ekvivalensek: $u \overset{Neigh}{\underset{r-1,p}{\sim}} v$*
  - *u-ból és v-ből multiplicitás szerint kerekítve ugyanannyi l címkéjű él vezet a $C \in \mathcal{N}/\overset{Neigh}{\underset{r-1,p}{\sim}}$ osztályú csúcsokba:*
    $\forall C \in \mathcal{N}/\overset{Neigh}{\underset{r-1,p}{\sim}}, \forall l \in Lab^E \Big($
    $mult_p(|\{e|\ src(e) = u\ \wedge\ lab^E(e) = l\ \wedge\ trg(e) \in C\}|) =$
    $mult_p(|\{f|\ src(f) = v\ \wedge\ lab^E(f) = l\ \wedge\ trg(f) \in C\}|)\Big)$
  - *u-ba és v-be multiplicitás szerint kerekítve ugyanannyi l címkéjű él vezet a $C \in \mathcal{N}/\overset{Neigh}{\underset{r-1,p}{\sim}}$ osztályú csúcsokból:*
    $\forall C \in \mathcal{N}/\overset{Neigh}{\underset{r-1,p}{\sim}}, \forall l \in Lab^E \Big($
    $mult_p(|\{e|\ trg(e) = u\ \wedge\ lab^E(e) = l\ \wedge\ src(e) \in C\}|) =$
    $mult_p(|\{f|\ trg(f) = v\ \wedge\ lab^E(f) = l\ \wedge\ src(f) \in C\}|)\Big)$

26. Definíció (Szomszédsági reprezentáció). *Egy adott $r \in \mathbb{N}$ és $p \in \mathbb{N}^+$ esetén az r távolságú p párhuzamosságú* szomszédsági reprezentáció *alatt az alábbi $NRep_{r,p}$ halmaz elemeit értjük:*

- *$r = 0$ esetén: $NRep_{0,p} = Lab^N$*
- *$r > 0$ esetén: $NRep_{r,p} = NRep_{r-1,p} \times 2^{NRepEntry_{r-1,p}} \times 2^{NRepEntry_{r-1,p}}$, ahol: $NRepEntry_{r-1,p} = Lab^E \times NRep_{r-1,p} \times M_p$*

27. Definíció (Reprezentáló függvény). *Legyenek $r \in \mathbb{N}$ és $p \in \mathbb{N}^+$. Ekkor a $\overset{Neigh}{\underset{r,p}{\sim}}$ ekvivalenciát reprezentálja az alábbi $nrep_{r,p} : \mathcal{N} \mapsto NRep_{r,m}$ függvény:*

$$nrep_{r,p}(n) \quad = \quad \begin{cases} lab^E(n) & ha\ r = 0 \\ (nrep_{r-1,p}(n), IN, OUT) & különben \end{cases}$$

ahol:

- *IN elemei olyan $(l,c,m) \in NRepEntry_{r-1,p}$ hármasok, amelyekre teljesül, hogy az olyan $(e,u) \in I \subseteq \mathcal{E} \times \mathcal{N}$ párok, amelyeknél $trg(e) = n$, $src(e) = u$, $lab^E(e) = l$ és $nrep_{r-1,p}(u) = c$ esetén $I \neq \emptyset$ és $m = mult_p(|I|)$.*

- *OUT elemei az előző ponthoz hasonlóan olyan* $(l, c, m) \in NRepEntry_{r-1,p}$ *hármasok, amelyekre teljesül, hogy az olyan* $(e, u) \in O \subseteq \mathcal{E} \times \mathcal{N}$ *párok, amelyeknél* $trg(e) = u$, $src(e) = n$, $lab^E(e) = l$ *és* $nrep_{r-1,p}(u) = c$ *esetén* $O \neq \emptyset$ *és* $m = mult_p(|O|)$.

D. Absztrakt gráftranszformáció

28. Definíció (Formatranszformáció). *Legyen P egy gráftranszformáció. A $t_P\mathcal{S} \mapsto 2^{\mathcal{S}}$ függvényt formatranszfromációnak nevezzük, ha $t_P(S) = SH$ esetén teljesül az alábbi feltétel:*

$$\forall G \in concr(S), \forall m, \exists F\Big(G \xrightarrow{P,m} F \;\Rightarrow\; shaping(F) \in SH\Big)$$

A transzformációkkal paraméterezhető t neve absztrakt transzformátor.

29. Definíció (Tökéletes absztrakt transzformátor). *A t transzfromátort tökéletesnek nevezzük, ha minden P gráftranszformáció esetén $t_P(S)$ minimális.*

30. Definíció (Absztrakt morfizmus). *Egy G gráf és egy S forma közötti* absztrakt morfizmuson *egy olyan m morfizmust értünk, amelyre teljesülnek az alábbi feltételek:*

- $\forall e \in E_G\Big(lab^E_G(e) = olab_S(m(e))\Big)$
- *minden G-beli l címkéjű csúcs összeegyeztethető $m(n)$-nel.*

31. Definíció (Fókuszált forma). *Legyen G egy címkézett gráf. Az F gráfot G-re* fókuszált formának *nevezzük, ha részgráfként tartalmazza G-t, G-n kívüli elemei egy formának megfelelően vannak címkézve és G-n kívül megfelel a forma feltételeinek, azaz:*

- *„Nincsenek G-n kívüli hasonló csúcsok":*

$$\forall n_1, n_2 \in N_F \setminus N_G\Big(n_1 \neq n_2 \Rightarrow comp_F(n_1) \neq comp_F(n_2)\Big)$$

- *„Nincsenek G-n kívüli párhuzamos élek":*

$$\forall e_1, e_2 \in E_F \setminus E_G\Big(e_1 \neq e_2 \Rightarrow \\ \neg\Big(olab_F(e_1) = olab_F(e_2) \wedge src_F(e_1) = src_F(e_2) \wedge trg_F(e_1) = trg_F(e_2)\Big)\Big)$$

32. Definíció (Fókusz). *Legyen G egy címkézett gráf, S egy forma, m_a egy G és F közötti absztrakt morfizmus, F egy G-re fókuszált forma. Akkor mondjuk, hogy S-nek m_a szerinti* fókusza *F, ha létezik olyan F és S közötti m_f* fókusz morfizmus *amelyre teljesülnek az alábbi állítások:*

- *S minden olyan g_S csúcsára és élére, amely nem szerepelt az absztrakt morfizmusban (azaz $\operatorname{inv} m_a(g_S) = \emptyset$, és él esetén még $\operatorname{inv} m_a(src_s(g_S)) = \emptyset \wedge \operatorname{inv} m_a(trg_s(g_S)) = \emptyset$), teljesül, hogy $\operatorname{inv} m_f(g_S) = \{g_F\}$ egyelemű halmaz, g_S absztrakt gráfelem, kompatibilitási és instrumentális reprezentációja illetve élcímkéje megegyezik g_F-évl.*
  Szemléletesen: fókuszálás során nem változnak az absztrakt morfizmusban nem érintett elemek.
- *S minden olyan g_S csúcsára és élére, amely szerepelt az absztrakt morfizmusban (azaz $\operatorname{inv} m_a(g_S) \neq \emptyset$, vagy él esetén még $\operatorname{inv} m_a(src_s(g_S)) \neq \emptyset \vee \operatorname{inv} m_a(trg_s(g_S)) \neq \emptyset$)), teljesülnek az alábbi feltételek:*

1. *Ha g_S egy él, és g_F egy konkrét él, akkor g_F címkéje meg kell hogy egyezzen $olab_S(g_S)$-bal.*
2. *Ha g_S egy él, és g_F is egy absztrakt él, akkor $olab_F(g_F) = olab_S(g_S)$.*
3. *Ha g_S egy csúcs, és g_F egy konkrét csúcs, akkor g_F címkéje csak olyan lehet, amely $comp_S(g_S)$-sel összeegyeztethető.*
4. *Ha g_S egy csúcs, és g_F is egy absztrakt csúcs, akkor $comp_F(g_F) = comp_S(g_S)$.*
5. *Minden g_S instrumentális ekvivalenciája összeegyeztethető a $\operatorname{inv} m_f(g_S)$ halmazzal.*

Szemléletesen: az absztrakt morfizmusban érintett elemek szétoszlanak G-nek megfelelő konkrét elemekké, illetve meghagyhatnak még olyan absztrakt elemeket, amelyek a maradékot képzik.

33. Definíció (Elérhető absztrakt állapot). *Legyen $S_0 \in \mathcal{S}$, R gráftranszformációs szabályok halmaza és t egy absztrakt transzformátor. Ekkor S_0-ból R-beli transzformációkat t-vel alkalmazva P-beli jólformáltsági kritériumokat betartva elérhető T forma (jelölésben: $S \overset{t,R,P}{\longrightarrow} T$), ha léteznek olyan $Prod_0, \ldots, Prod_n \in R$ transzformációk, és $S_1, \ldots S_n \in \mathcal{S}$ formák, hogy:*

$$P_0, \ldots, P_n \text{ mindegyike összeegyeztethető a jólformáltsági kritériumokkal, és}$$

$$S_1 \in t_{P_0}(S_0), S_2 \in t_{P_1}(S_1), \ldots, T \in t_{P_n}(S_n)$$

Egy gráf önmagából definíció szerint elérhető.

34. Definíció (Absztrakt állapottér). *Legyen G_0 címkézett kiinduló gráf, R gráftranszformációs szabályok halmaza és t egy absztrakt transzformátor. Ekkor az $absstates(G_0, R, t)$ függvény eredménye egy olyan $ABSSTATES \in \mathcal{G}(\mathcal{S}, R)$ címkézett gráf, amelyre az alábbi állítások igazak:*

- $S_0 = shaping(G_0)$
- *Minden csúcscímke csak egyszer szerepel, más szóval $lab^N_{ABSSTATES}$ injektív.*
- *Csúcsai azon S formákkal vannak címkézve, amelyekre teljesül, hogy $S \overset{t,R,S_0}{\longrightarrow}$.*
- *Minden olyan S és T formához és $P \in R$ transzformációhoz, amelyre teljesül, hogy $S \overset{t,R,S_0}{\longrightarrow}$, $T \overset{t,R,S_0}{\longrightarrow}$ és $T \in t_P(S)$ pontosan egy él tartozik. Ennek az élnek a címkéje P, irányát tekintve a S-sel címkézett csúcsból vezet a T-vel címkézett csúcsba.*

E. Következtető rendszerek

35. Definíció (Multiplicitás összeg). *Egy adott $m \in \mathbb{N}^+$ esetén legyen $+ : M_m \times M_m \mapsto M_m$ az alábbi módon definiált függvény:*

$$a + b = \begin{cases} \omega & ha\ a = \omega\ vagy\ b = \omega \\ mult_m(a + b) & különben \end{cases}$$

36. Definíció (Multiplicitás levonás). *Egy adott $m \in \mathbb{N}^+$ esetén legyen $minus : M_m \mapsto 2^{M_m}$ az alábbi módon definiált függvény:*

$$minus(a) = \begin{cases} \emptyset & ha\ a = 0 \\ \{a - 1\} & ha\ a > 0 \\ \{m, \omega\} & ha\ a = \omega \end{cases}$$

37. Definíció (Pontosan 1 teljesülés). *Legyenek $F_1, \dots F_n$ logikai formulák. Ekkor:*

$$\bigoplus\{F_1, \dots F_n\} \Leftrightarrow \bigvee_{i=1}^{n} \bigwedge_{j=1}^{n} (F_j \wedge i = j)$$

38. Definíció (Multiplicitás kvantor). *Legyenek $\exists^{\geq m} XF(X)$ egy állítás. Ennek jelentése legyen a következő:*

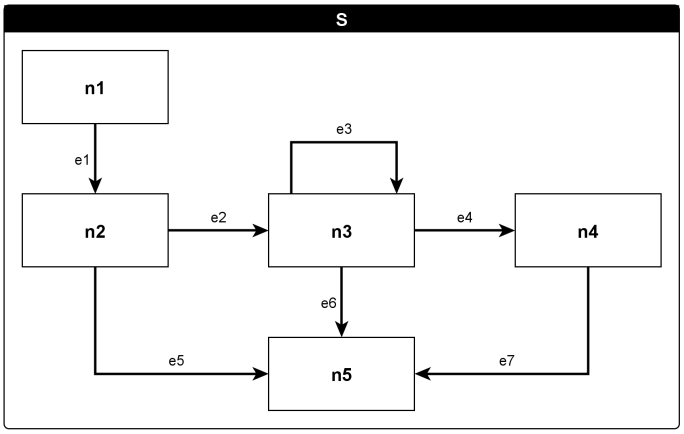
$$\exists^{\geq m} XF(X) \Leftrightarrow \exists X_1 \dots X_m \Big(\bigwedge_{i=1}^{m} F(X) \wedge \bigwedge_{j=1, i \neq j}^{m} (X_i \neq X_j) \Big)$$

Továbbá $\exists^{=m} XF(X)$ azt jelentse, hogy

$$\exists^{=m} XF(X) \Leftrightarrow (\exists^{\geq m} XF(X)) \wedge \neq (\exists^{\geq m+1} XF(X))$$

---

23. Példa Legyen `S` az alábbi struktúrájú forma:



Az ennek megfelelő gráfstruktúrából és szürjektív morfizmusból származó axiómák a következő oldalon találhatóak. A `%`-kal kezdődő sorok kommenteket tartalmaznak, ezeket természetesen nem dolgozza fel a program.

```
%
% Gráf definíciók
%
% Gráf struktúra definíciók:
all X ((g_Node(X) & -g_Edge(X)) | (-g_Node(X) & g_Edge(X))).
all E (g_Node(getSource(E)) & g_Node(getTarget(E))).
all N (g_Node(N)-> (
(lab_N_Cell(N) & -lab_N_List(N) & -lab_N_Object(N))|
(-lab_N_Cell(N) & lab_N_List(N) & -lab_N_Object(N))|
(-lab_N_Cell(N) & -lab_N_List(N) & lab_N_Object(N)))).
all E (g_Edge(E)-> (
(lab_E_first(E) & -lab_E_next(E) & -lab_E_value(E))|
(-lab_E_first(E) & lab_E_next(E) & -lab_E_value(E))|
(-lab_E_first(E) & -lab_E_next(E) & lab_E_value(E)))).
% Jólformáltsági kritériumok (0)

%
% szürjektív morfizmus axiómái:
%
% A morfizmus egzisztenciális
exists X s_N_n1(X).
exists X s_N_n2(X).
exists X s_N_n3(X).
exists X s_N_n4(X).
exists X s_N_n5(X).
exists X s_E_e1(X).
exists X s_E_e2(X).
exists X s_E_e3(X).
exists X s_E_e4(X).
exists X s_E_e5(X).
exists X s_E_e6(X).
exists X s_E_e7(X).
% A morfizmus teljes leképezés
all X (g_Node(X) -> (
(s_N_n1(X) & -s_N_n2(X) & -s_N_n3(X) & -s_N_n4(X) & -s_N_n5(X)) |
(-s_N_n1(X) & s_N_n2(X) & -s_N_n3(X) & -s_N_n4(X) & -s_N_n5(X)) |
(-s_N_n1(X) & -s_N_n2(X) & s_N_n3(X) & -s_N_n4(X) & -s_N_n5(X)) |
(-s_N_n1(X) & -s_N_n2(X) & -s_N_n3(X) & s_N_n4(X) & -s_N_n5(X)) |
(-s_N_n1(X) & -s_N_n2(X) & -s_N_n3(X) & -s_N_n4(X) & s_N_n5(X)))).
all X (g_Edge(X) -> (
(s_E_e1(X) & -s_E_e2(X) & -s_E_e3(X) & -s_E_e4(X) &-s_E_e5(X) &
-s_E_e6(X) & -s_E_e7(X)) |
(-s_E_e1(X) & s_E_e2(X) & -s_E_e3(X) & -s_E_e4(X) & -s_E_e5(X) &
-s_E_e6(X) & -s_E_e7(X)) |
(-s_E_e1(X) & -s_E_e2(X) & s_E_e3(X) & -s_E_e4(X) & -s_E_e5(X) &
-s_E_e6(X) & -s_E_e7(X)) |
(-s_E_e1(X) & -s_E_e2(X) & -s_E_e3(X) & s_E_e4(X) & -s_E_e5(X) &
```

```
-s_E_e6(X) & -s_E_e7(X)) |
(-s_E_e1(X) & -s_E_e2(X) & -s_E_e3(X) & -s_E_e4(X) & s_E_e5(X) &
-s_E_e6(X) & -s_E_e7(X)) |
(-s_E_e1(X) & -s_E_e2(X) & -s_E_e3(X) & -s_E_e4(X) & -s_E_e5(X) &
s_E_e6(X) & -s_E_e7(X)) |
(-s_E_e1(X) & -s_E_e2(X) & -s_E_e3(X) & -s_E_e4(X) & -s_E_e5(X) &
-s_E_e6(X) & s_E_e7(X)))).
% Csúcsokat csúcsokhoz, éleket élekhez
all X (s_N_n1(X) -> g_Node(X)).
all X (s_N_n2(X) -> g_Node(X)).
all X (s_N_n3(X) -> g_Node(X)).
all X (s_N_n4(X) -> g_Node(X)).
all X (s_N_n5(X) -> g_Node(X)).
all X (s_E_e1(X) -> g_Edge(X)).
all X (s_E_e2(X) -> g_Edge(X)).
all X (s_E_e3(X) -> g_Edge(X)).
all X (s_E_e4(X) -> g_Edge(X)).
all X (s_E_e5(X) -> g_Edge(X)).
all X (s_E_e6(X) -> g_Edge(X)).
all X (s_E_e7(X) -> g_Edge(X)).
% Források és célok definiálása
all E (s_E_e1(E) -> s_N_n1(getSource(E))).
all E (s_E_e1(E) -> s_N_n2(getTarget(E))).
all E (s_E_e2(E) -> s_N_n2(getSource(E))).
all E (s_E_e2(E) -> s_N_n3(getTarget(E))).
all E (s_E_e3(E) -> s_N_n3(getSource(E))).
all E (s_E_e3(E) -> s_N_n3(getTarget(E))).
all E (s_E_e4(E) -> s_N_n3(getSource(E))).
all E (s_E_e4(E) -> s_N_n4(getTarget(E))).
all E (s_E_e5(E) -> s_N_n2(getSource(E))).
all E (s_E_e5(E) -> s_N_n5(getTarget(E))).
all E (s_E_e6(E) -> s_N_n3(getSource(E))).
all E (s_E_e6(E) -> s_N_n5(getTarget(E))).
all E (s_E_e7(E) -> s_N_n4(getSource(E))).
all E (s_E_e7(E) -> s_N_n5(getTarget(E))).
```

Irodalomjegyzék

[1] J. Warmer A. Kleppe and W. Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise.* Addison-Wesley, 2003.

[2] Pataricza András, editor. *Formális módszerek az informatikában.* Typotex, 2006.

[3] Patrick Cousot and Radhia Cousot. Systematic design of program analysis frameworks. Symposium on Principles of Programming Languages, 1979.

[4] D. McGuinness D. Nardi F. Baader, D. Calvanese and editors P. F. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications.* Cambridge University Press, 2003.

[5] István Ráth Dániel Varró Gábor Bergmann, Zoltán Ujhelyi. A Graph Query Language for EMF models. *Transformations, Fourth International Conference, ICMT 2011, Zurich, Switzerland, June 27-28, 2011. Proceedings. Volume 6707 of Lecture Notes in Computer Science, pages 167-182*, 2011.

[6] H.-J. Kreowski H. Ehrig, G. Engels and editor G. Rozenberg. *Handbook on Graph Grammars and Computing by Graph Transformation.* World Scientific, 1999.

[7] I. Horrocks and S. Tessari. Querying the semantic web: a formal approach. *Proc. of the 13th Int. Semantic Web Conf. ISWC 2002, number 2342 in Lecture Notes in Computer Science*, 2002.

[8] Marcos E. Kurbán Iovka Boneva, Arend Rensink and Jörg Bauer. Graph Abstraction and Abstract Graph Transformation. 2007.

[9] J.Sztipanovits and G.Karsai. *Model-integrated computing.* IEEE Computer, 30(4), 1997.

[10] Stephen Cole Kleene. *Introduction to Metamathematics.* Ishi Press International, 1987.

[11] Thomas Reps Mooly Sagiv and Reinhard Wilhelm. Parametric Shape Analysis via 3-Valued Logic. *ACM Transactions on programming Languages and Systems, Vol. 24, No. 3*, 2002.

[12] Arend Rensink. A Logic of Local Graph Shapes. *CTIT Technical Report TR-CTIT-03-35*, 2003.

[13] Arend Rensink. Canonical Graph Shapes. *ESOP 2004, LNCS 2986*, pages 401–415, 2004.

[14] Arend Rensink and Dino Distefano. Abstract Graph Transformation. *International Workshop on Software Verification and Validation (SVV)(2005), Electronic Notes in Theoretical Computer Science*, 2005.

[15] Budapesti Műszaki és Gazdaságtudományi Egyetem. Viatra2 model transformation framework. `http://www.eclipse.org/gmt/VIATRA2`.

[16] cumputer science department The University of New Mexico. Prover9 and Mace4, howpublished=`http://www.cs.unm.edu/mccune/mace4/`,.

[17] Dániel Varró and András Balogh. The model transformation language of the VIATRA2 framework. *Science of Computer Programming, 68(3)*, 2007.

[18] yWorks. yEd Graph Editor. `http://www.yworks.com`.