Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Nagyméretű modelltranszformációk megvalósítása relációs adatbázisok fölött

Horváth Dóra (DOP0IA), I. évf, (MSc) mérnök inf. szakos hallgató
Konzulens: Bergmann Gábor és Horváth Ákos, MIT
Szolgáltatásbiztos rendszertervezés szakirány
Önálló laboratórium 1 összefoglaló
2009/10. II. félév

A modellvezérelt szoftverfejlesztés napjaink egyik felfutóban lévő paradigmája. Ennek során a fejlesztők az elkészítendő rendszer egy magas szintű és általános modelljéből indulnak ki és modelltranszformációs lépések során jutnak el az alkalmazás futtatható változatáig. A modelltranszformáció modellek egymásba képzését jelenti, amely során fontos, hogy nagyméretű modelleken is gyorsan lefussanak. A nagyméretű, többszázezer csomópontból és kapcsolatból álló modellek esetén egy kritikus kérdés a tárolás. Egy-egy modellelem összetett (Java) objektumként való tárolása memóriakorlátokba ütközik, ezért az iparban bevált módszer, hogy adatbázisban tárolják a modelleket. Ebben az esetben azonban a kellő gyorsaságot nem tudják produkálni modelltranszformációknál, míg a memóriában lévő objektumokkal (kisebb modelleken) igen. Erre a problémára megoldást nyújthatnak a memóriába ágyazott adatbázisok. Az önálló labor témámban ezt vizsgáltam meg.

A modelltranszformációt két lépésre lehet bontani a mintaillesztésre és a modellmanipulációra. A mintaillesztés során gráfmintákat keresünk modellen, a manipuláció során pedig az erre illeszkedő részgráfokon bizonyos műveleteket végzünk. A mintaillesztés egyik fajtája az inkrementális mintaillesztés, ez esetben az illeszkedő részgráfokat cache-ben tároljuk, amit folyamatosan karban kell tartani. A karbantartás a modell különböző változásaikor történik, a változásokról pedig rendszer küld értesítéseket. A relációs adatbázisok körében ezt az úgynevezett triggerek segítségével lehet megtenni.

Az előző féléves szakdolgozatom során a MIT-en fejlesztett VIATRA2 modelltranszformációs keretrendszerhez egy plugint fejlesztettem, ami rendszerbe betöltött gráfmintákból triggereket automatikusan származtat. Tehát az alkalmazás az inkrementális mintaillesztést támogatja. A szakdolgozat végén megfogalmazott továbblépési lehetőségek közül valósítottam meg néhányat ebben a félévben. Így az alkalmazás képes bonyolultabb gráfminták leképzésére is, mint például a kört tartalmazó minták illetve az injektivitást is kezeli. A triggerek szerkezetükben is megváltoztak. A korábbiakban a mintára illeszkedő részgráfokat az új/töröl modellelem szűk környezetének kurzoros bejárásával kereste meg a trigger. Az új változatban pedig a gráfminta leírás szélességi bejárása alapján elkészített join lekérdezés segítségével történik a minták illesztése. A join használatát az teszi indokolttá, hogy ezt a műveletet az adatbázis lekérdezés optimalizálója elemezi és optimalizálja, végül több tényező alapján a rendelkezésre álló join algoritmusok közül választ. Ezzel szemben a kurzor csak egy adott join algoritmust valósít meg.

Méréseimet a Petri-háló tranzíciós rendszeren végeztem el. A mérések során a háló egy tüzelésének idejét mértem meg, különböző méretű Petri-hálók esetén. Az eredményeket egy másik mintaillesztési eljárással, a keresés alapúval hasonlítottam össze. (A keresés alapú mintaillesztési algoritmus során a mintákat a teljes modell adott pontból indított bejárásával keressük meg.) A következő megállapításokat lehet tenni az eredmények alapján: az inkrementális mintaillesztés két nagyságrenddel gyorsabb a keresés alapúnál – bizonyos esetekben. Míg a keresés alapú mintaillesztés esetén a mért egység, azaz az egy tüzelés ideje a Petri-háló méretével azonos arányban nőtt, az inkrementális mintaillesztés esetén sokkal kevesebbre adódott. Ezek az eredmények függnek a Petri-háló kezdeti token eloszlásától.