THE PROLOG WITH C++ INTERCONNECTION LIBRARY

Petra HRABCOVÁ, Master Degree Programme (5)
Dept. of Intelligent Systems, FIT, BUT
E-mail: xhrabc02@stud.fit.vutbr.cz

Supervised by: Ing. Martin Hrubý

ABSTRACT

This text deals with design and implementation of a library which interconnects two programming languages, Prolog and C++. The library allows using expressing style of Prolog language in applications based on C++ language. The whole software package supports so called heterogeneous modeling.

1 ÚVOD

V různých oblastech se setkáváme s úlohami, které lze snadno řešit některým druhem programovacích jazyků a pro jiné jsou velkým problémem. Jazyky, které nazýváme deklarativní, jsou vhodné například pro řešení problému umělé inteligence, neboť programátor popisuje daný problém pomocí tzv. predikátů a výsledek je odvozen ze známých faktů. Takovým jazykem je například Prolog.

Naopak problémy, jimž lépe vyhovuje sekvenční postup řešení, zpracováváme zpravidla v jazycích imperativních, jako je například C++ nebo Visual Basic.

Jak ale postupovat v případě, že úloha vyžaduje kombinaci obou těchto přístupů? Existují jazyky, které oba přístupy dovolují kombinovat, např. Lisp, případně tzv. multi-paradigm jazyky, mezi jejichž zástupce se řadí jazyk OLI [1]. V některých případech může být ale přínosnější použití jazyka čistě imperativního a čistě deklarativního, které spolu budou umět komunikovat. Právě komunikace mezi logickým jazykem Prolog a imperativním jazykem C++ se stala náplní mého semestrálního projektu.

2 PROPOJENÍ JAZYKA C++ A JAZYKA PROLOG

Jelikož výsledný software bude použit pro simulační účely, konkrétně pro heterogenní modelování [2] a v jazyce C++ bude možné spolupracovat se simulační knihovnou SIMLIB/C++, byla zvolena kombinace právě těchto dvou jazyků.

SIMLIB/C++ je experimentální objektově orientovaná simulační knihovna vhodná pro modelování a simulaci v jazyce C++. Obsahuje základní prostředky pro vytváření spojitých, diskrétních, kombinovaných a heterogenních simulačních modelů.[3]

Při zkoumání možností, které pro propojení zmíněných jazyků existují, jsem nalezla knihovnu, která je součástí implementace jazyka SWI-Prolog [4] a tvoří základní prostředek pro komunikaci s jazykem C. Problémem této knihovny je však složitost jejího použití. Vzhledem k účelu tohoto projektu, tedy orientaci na simulaci a propojení s knihovnou SIMLIB/C++, bylo dalším nevhodným rysem neobjektové zpracování této knihovny.

Po zvážení možností byla navržena knihovna, která je založena na již zmíněné knihovně pro propojení SWI-Prologu a jazyka C, přidává objektovou orientaci a také zjednodušení použití oproti knihovně původní.

3 NÁVRH

Knihovna dovoluje volání kódu napsaného v Prologu z programu v jazyce C++. Její návrh byl rozdělen do tří částí. V první z nich byl navržen systém práce s datovými typy, které SWI-Prolog poskytuje a jejich převod do typů, jež jsou poskytovány jazykem C++. Všechny třídy reprezentující tyto typy mají stejné rozhraní, aby bylo možné ke všem typům přistupovat stejným způsobem.

Druhá část byla zaměřena na práci s těmito daty pomocí faktů a predikátů. Byla vytvořena třída Fact, která poskytuje všechny metody nutné k práci s fakty. Vše pak bylo propojeno poslední částí knihovny, která se zabývá stálou dostupností všech dat a jejich přiřazením k faktům a predikátům.

3.1 PŘÍKLAD

Rozdíly mezi oběma knihovnami (původní knihovnou a knihovnou vytvořenou jako součást mého semestrálního projektu) jsou patrné na následujícím příkladu.

Jedná se o vložení faktu otec(adam,jan) do databáze Prologu. V prvním případě je použita původní knihovna, druhém případě je použita knihovna, která byla vyvinuta jako součást tohoto projektu.

PŮVODNÍ KNIHOVNA

```
predicate_t pred = PL_predicate("asserta",1,“database“);
term_t term =PL_new_term_ref();
term_t terms = PL_new_term_refs(2);
PL_put_term(terms, PL_new_atom(„adam“));
PL_put_term(terms +1, PL_new_atom(„jan“));
atom_t name = PL_new_atom(„otec“);
functor_t func = PL_new_functor(name,2);
PL_cons_functor_v(term,func,terms);
qid_t result = PL_open_query(0,PL_Q_NORMAL,pred,term);
while(PL_next_solution(result)){}
PL_close_query(result);
```

NOVÁ KNIHOVNA

Nejprve je vytvořena třída, která reprezentuje veškeré predikáty arity 2, jejichž parametry jsou atomy. Poté je vytvořena její instance, která již reprezentuje konkrétní predikát.

```
class MyFact: public Fact
{public:
        MyFact(std::string aName): Fact(aName), osoba1(*this, 0), osoba2(*this,1){}
        StringData osoba1;
        StringData osoba2;
};
PrologConnection conn("database",0,0);
MyFact otec("otec");
otec.osoba1.setValue("adam");
otec.osoba2.setValue("jan");
otec.insert(&conn);
```

4 ZÁVĚR

Knihovna propojující deklarativní a imperativní jazyk může být jistě velikým přínosem. Jak je patrné z předchozího příkladu, použití původní knihovny pro propojení SWI-Prologu a jazyka C je pro praktické použití složité. Vytvořením objektově orientované knihovny, která sice neposkytuje všechny možnosti původní knihovny, ale přispívá k pohodlnějšímu využití propojení obou jazyků, bych chtěla přispět k rozvoji heterogenního modelování.

V současné době probíhají experimenty na prototypu knihovny, který dovoluje pracovat s několika datovými typy a dovoluje přidávat fakty za běhu aplikace. Časová náročnost použití prostředků jazyka Prolog z jazyka C++ je, dle provedených experimentů, vyhovující.

K praktickému využití této knihovny zbývá ještě implementovat další funkčnosti (zejména pro práci se seznamy), jež jsou poskytovány Prologem. Otevřenou otázkou také zůstává práce s predikáty (například vkládání nových, rušení atd.) za běhu aplikace.

Všemi těmito nedostatky bych se chtěla zabývat ve své diplomové práci, která by rovněž měla obsahovat aplikaci, která by funkčnost a možnosti této knihovny prezentovala.

LITERATURA

[1] Lee J.H.M., Pun P.K.C.: An Overview of the OLI Multiparadigm Programming Language and Its Semantics, 1996, http://citeseer.ist.psu.edu/lee96overview.html (únor 2006)

[2] Hrubý M., Kočí R., Rábová Z.: The Heterogeneous Modelling Methods and Tools, Simulation Almanac 2005, Praha, CZ, FEL ČVUT, 2005, s. 40-50

[3] Peringer P.: SIMulation LIBrary for C++, http://www.fit.vutbr.cz/~peringer/SIMLIB

[4] Wielemaker J.: Dept. of Social Science Informatics, Amsterdam. SWI-Prolog Reference manual, http://www.cse.psu.edu/~catuscia/teaching/prolog/Manual/Title.html (2006)