2007 年 12 月 19 日

2007年度後期・数理解析・計算機数学III プログラミング課題

評価方法について

今回のプログラミング課題と後に配るレポート課題を元にして行う．プログラムにはコメントを含めなければならない．

レポートの提出方法

プログラムソースコードは電子メールの添付ファイルとして提出すること．

提出期限

提出期限は 2006 年 1 月 22 日 (火).

課題

述語論理の項，述語およびホーン節を以下のように定義する．

```
type term =
  | V of string                                    (* 変数 *)
  | C of string                                    (* 定数 *)
  | F of term list                                 (* 関数 *)

type pred = string * term list
type clause = H of pred * pred list
type subst = (string * term) list
```

簡単な Prolog のインタープリタを作るために，以下の関数を定義せよ．

1. 項の代入および項の単一化
   ```
   val subst_term : subst -> term -> term
   val unify : subst -> (term*term) list -> subst
   ```
2. 7 章で見た merge を参考にして，二つのリストを重複のないリストにまとめる関数を定義せよ．
3. ホーン節の変数を重複なく集める
   ```
   val vars : clause -> string list
   ```

4. 述語とホーン節の代入
   `val subst_pred : subst -> pred -> pred`
   `val subst_clause : subst -> clause -> clause`

5. ホーン節の中の変数を新しいものに変える
   `val instance : clause -> clause`
   そのために以下の関数 `fresh_vars` を使うといい.

```
let genvar : string -> term =
  let counter = ref 0 in
  fun s -> incr counter; V (s ^ "/" ^ string_of_int !counter)
let fresh_vars : string list -> subst =
  List.map (fun x -> (x, genvar x))
```

6. ある述語に対して，それを帰結として持つ全てのホーン節の仮定を返す関数
   `val expand : clause list -> subst -> pred -> (subst * pred list) list`
   結果は連言の選言を表している (選言標準形に当る). `instance` および `unify` を使う必要がある. `flat_map` も役に立つ.

7. 代入と述語の連言をもらい，それを完全に解く代入のリストを返す関数.
   `val solve : clause list -> subst -> pred list -> subst list`

以下のデータに対してテストできる.

```
let clauses =
  [H(("grandfather",[V"x";V"y"]),
   ["father",[V"x";V"z"]; "father",[V"z";V"y"]]);
   H(("grandfather",[V"x";V"y"]),
   ["father",[V"x";V"z"]; "mother",[V"z";V"y"]]);
   H(("father",[C"Namihei";C"Sazae"]),[]);
   H(("father",[C"Namihei";C"Katsuo"]),[]);
   H(("father",[C"Namihei";C"Wakame"]),[]);
   H(("mother",[C"Sazae";C"Tarao"]),[])]

# solve clauses [] ["grandfather",[C"Namihei"; V"x"]];;
- : subst list =
[[("x", C "Tarao"); ("z/3", C "Sazae"); ("y/2", C "Tarao");
  ("x/1", C "Namihei")]]
```