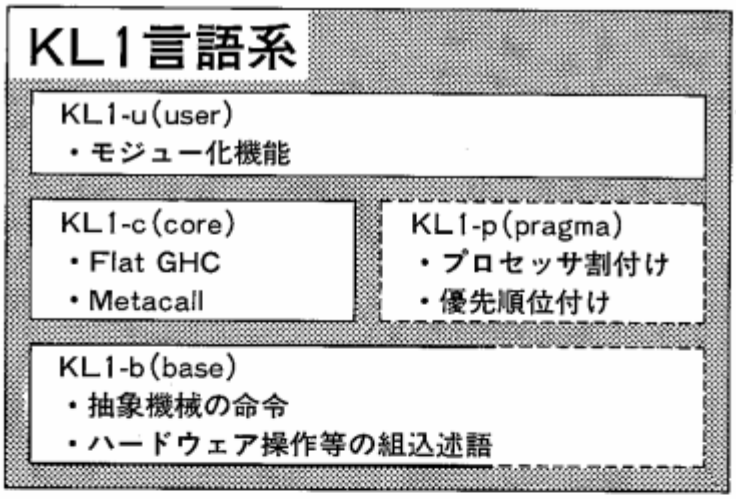


Guarded Horn Clausesとは

・構文

GHC ＝ Horn節 ＋ ガード

・意味

GHC ＝ Horn節 ＋ 因果性
＋ 非決定的選択

（因果性：情報の流れに関する制限）

cf. Prolog ＝ Horn節 ＋ 逐次制御
＋ ・・・

Guarded Horn Clausesの設計指針

基本指針：

(1) 逐次言語を拡張した並列言語ではなく，根本的に並列言語であること。
ただし，処理系は，プログラムの意味を変えない限り逐次処理を行なってよい。

(2) 並列アルゴリズムを記述できる汎用言語であること。

(3) 単純であること。

(4) 効率的な実現の見通しがあること。

より具体的な指針：

(5) 論理型(関係型)プログラミングの枠組に基くこと。

(6) 計算プロセスと外界を統一的に扱えること。
つまり，入出力とプロセス間通信を，同一の機構で行なうこと。

Guarded Horn Clausesの構文

- プログラム——プログラム節の集合
- プログラム節

$$H :- G_1, \cdots, G_m \mid B_1, \cdots, B_n.$$

組込述語“=”：X＝X．（通常のHorn節）

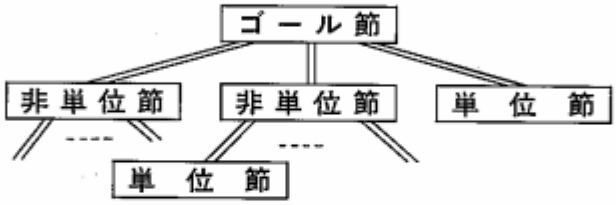
- ゴール節

$$:- B_1, \cdots, B_n.$$ （通常のHorn節）

Guarded Horn Clausesの意味

基本：並列入力導出

証明木（AND木）：



- 証明木の枝の集合は，単一化問題（連立方程式）を定める。
- 証明木は複数個ありうる（OR並列性）。

証明：次のことを並列に行なう。

(1) 可能な証明木を作る。

(2) 個々の証明木の定める単一化問題を解く。

Guarded Horn Clausesの意味

制限：単一環境

H_1：— …, G_{1j}, … | …, B_{1j}, …

H_2：— …, G_{2j}, … | …, B_{2j}, …

⋮ ⋮ ⋮

（ガード）　（ボディ）

ガードは並列に試みてよいが呼び出し側を具体化できない

呼び出し側を具体化できるが唯一つの選択された節に限る

カードが成功した節からひとつ選択する（コミット）

プログラム例

ストリームの併合

```
merge([A | X'], Y, Z) :- true |
  Z=[A | Z'], merge(X', Y, Z').
merge(X, [A | Y'], Z) :- true |
  Z=[A | Z'], merge(X, Y', Z').
merge( [], Y, Z) :- true | Z=Y.
merge( X, [], Z) :- true | Z=X.
```

プログラム例

整数列生成器

・データ駆動（自律的）

```
gen(N, Ns) :- true |
  N' :=N+1, Ns=[N' | Ns'], gen(N', Hs').
```

・要求駆動（他律的）

```
gen(N, [M | Ns']) :- true |
  N' :=N+1, M=N', gen(N', Ns').
```

プロセス記述言語としてのGHC

- (プロセス)系：　証明木の中のゴール群
- プロセス：　ゴール(の実行過程)
- プロセスの"状態"：(サブ・ゴールの引数が保持)
- 計算：　単一化とゴールのリダクション
  (bindingの観測と生成)
- 通信：　ゴール間共有変数に対する単一化
- 同期：　ガードの意味規則による単一化の中断
- 非決定的選択：　コミット

- プロセスのひな型：述語定義

Guarded Horn Clausesの特徴

他の論理型言語と比較して

(1) きわめて単純である

(2) 単一環境の制約の下で，論理プログラミングの枠組を最大限に保存している

他の並列言語と比較して

(1) きわめて単純である

(2) 次のような概念を実現するための一般的かつ統一的な枠組を提供する：

- データ駆動／要求駆動計算
- "lenientcons"
- 書き換え可能な配列／データベース
- 差分プログラミング
- Non-strictなデータ構造
- 動的プロセス／データ構造
- オブジェクト指向プログラミング