# 1J－9 一階論理コンパイラを用いる分散定理証明システムの実装と性能評価

芦澤宏樹　茅野康臣　岩沼宏治
山梨大学工学部電子情報工学科

## 1 概要

Stickel [1] により開発された Prolog Technology Theorem Prover (**PTTP**) は一階論理コンパイラであり，充足不可能であることを証明すべき一階論理式が与えられると，その上のトップダウン型の演繹を模倣する高速な Prolog プログラムを出力する．

本研究では，より高速な定理証明システムの構築を目指し，分散処理型の Prolog コードを生成する一階論理コンパイラを構築する．生成されるコードは，LAN で結合された WS 群の上で実行される．オーバーヘッドを軽減するため，分散処理の基本形態は極めて単純なものを採用し，幾つかの問題について性能評価実験を行なった．

## 2 PTTP

PTTP は，充足不可能であることを証明すべき一階論理式が与えられると，その論理式集合からの Model Elimination ( **ME**) 演繹 [2] 及び，Depth-First Iterative-Deepening (**DFID**) 探索戦略 [3] を模倣した定理証明を行なう **Prolog プログラム**に変換し，それを既存の Prolog 処理系の上で実行することにより定理証明を行なうシステムである．コンパイラにより出力される Prolog コードはそれ自身極めて高速に動作するので，高速な定理証明を行なうことが出来る．

## 3 分散処理

### 3.1 各プロセス間の通信

本研究では，PTTP の出力コードの実行を LAN で結合された WS 群の上で分散処理することにより，より高速な定理証明の実現を目指す．各プロセス間の通信手段は，Sicstus Prolog に標準で附属している **Linda** [4] のライブラリを用いて行なった．

A Distributed Theorem Prover with First-order Logic Compiler: an Implementation and Experiments
Hiroki Ashizawa　Yasuomi Chino　Kouji Iwanuma
Yamanashi University
4-3-11 Takeda, Koufu, Yamanashi 400, Japan

### 3.2 分散定理証明アルゴリズム

各 WS 間の結合は疎結合であるので，通信によるコストは大きなものになる．従って，WS 間の通信回数を減らすことが効率化に大きく貢献する．今回はオーバーヘッドの軽減を考慮し，以下に示すような単純な分散定理証明アルゴリズムで実験を行なった．

1. プロセスは一個の**親プロセス**と一個以上の**子プロセス**の 2 種類を用意する．それぞれは論理式の集合を，Prolog プログラムとしてあらかじめ持つほか，親プロセスは質問節を持つものとする．
2. まず親プロセスが質問節から導出を始める．
3. 親プロセスが DFID による深さ制限内の探索を終えた時点で，導出された節の数が子プロセスの数より多い時，それらを子プロセスに分配し，続きは子プロセスに解かせる (4へ)．導出された節の数が子プロセスより小さい時は，親プロセスが次の深さ制限内を解く (3へ)．
4. 子プロセスは，親プロセスが解いた次の深さ制限からその続きを DFID を使って解き始める．

例として，導出木が完全 2 分木で子プロセスが 3 個の場合の親プロセスと子プロセスの探索空間の分担を図 1に示す．

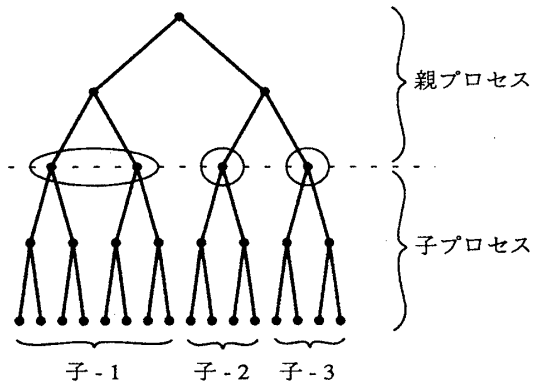


図 1: 子プロセスが 3 個の場合

このアルゴリズムはProlog のor並列性を利用している．そのため変数などを共有する必要がなく，親プロセスから子プロセスに導出された節を渡す時のみにプロセス間の通信は行なわれ，最低限の通信回数で分散することが可能である．

しかし逆に，プロセス間の通信を最低限しか行なわないため，各子プロセスにかかる負荷にばらつきがあることや，ある時刻において各子プロセスが解いている深さ制限が一定でないことなどの短所も生じる．

## 4 実験結果

今回の実験は，PTTP を用いずに我々が独自に開発したコンパイラを改造して行なった．実行にはSun SPARC staion 2 上で Sicstus Prolog コンパイラを使用した．図2〜図4に示すグラフは，横軸にプロセスの数をとり，縦軸には，分散処理を行なわずに証明を行なった場合の時間を各プロセスの数において解くのに要した時間で割った比をとった．

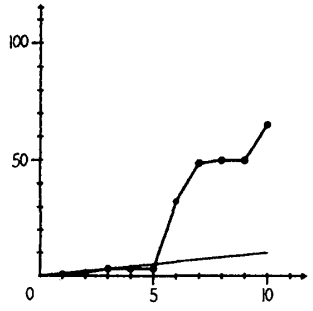


図 2 ls36

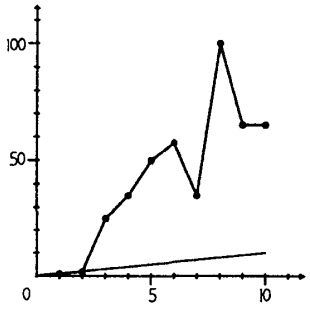


図 3 wos4

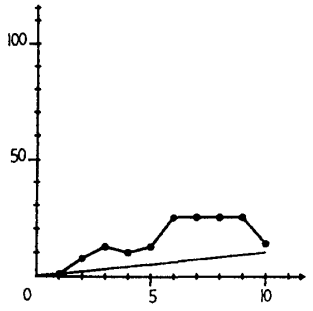


図 4 apabhp

表 1 実行時間 [sec]

| 問題 | 時間 |
|---|---|
| ls36 | 551.8 |
| wos4 | 4557.1 |
| apabhp | 1063.7 |

また，表1は，分散を行なわずに証明を行なった場合に要した実行時間である．

## 5 考察

前章の実験結果より，プロセスの数を増やしていくと線形台数効果以上の高速化が確認できる．これは，同じ深さ制限内において，各子プロセスが探索している空間の広さにばらつきがあるせいと思われる．探索空間が小さいプロセスは同じ時間内により長い証明を見つけることが出来る．一般に定理の証明パスは複数存在するので，探索空間が小さいところに解があると早く証明が終了することができ，前章で示したような結果が得られたものと考えられる．従って，絶対的な実行時間は，当初の目標よりかなり良い結果を得ることが出来た．しかし，その探索空間のばらつきによる問題が，いくつか考えられる．

1. プロセスの数を増やしても，必ずしも増やす前の実行速度より速くなるとは限らない．図3, 図4からわかる通り，プロセスの数を増やすことにより，証明を見つける時間が長くなることがあり得る．
2. 今回は一つの証明を見つけるまでの時間を求めたので，上記のばらつきにより高速な定理証明を行なうことが出来た．しかし，全解探索を行なうような場合，このアルゴリズムでは線形代数効果以上の効果を得ることはありえない．

現在は，オーバーヘッドをなるべく増やさずに，各プロセスにかかる負荷をもう少し均等にして上の二つの問題を解決する為の手法と，**補題**を組み込むことによる冗長な再計算の削除を行ない，より高速に定理証明を行なうための手法を検討中である．

## 参考文献

[1] M.E. Stickel, A prolog technology theorem prover: a new exposition and implementation in prolog, *Thoretical Computer Science* **104**(1992) 109-128.

[2] S. Fleisig, D .Loveland, A.K. SmileyIII and D.L. Yarmush, An implementation of the model elimination proof procedure. *Journal of the ACM* **21** (1) (1974) 124-139.

[3] R.E. Korf, Depth-first iterative-deepening: an optimaladmissible tree search, *Artijicial Intelligence*, 1985.

[4] Carriero, N., and Gelernter, D., Linda in context. *Communications of the ACM***32** (Apr. 1989) 444-458.