1J－8

トップダウン型定理証明における補題の有用性
－その実装と評価－

茅野 康臣　芦澤 宏樹　岩沼 宏治
山梨大学工学部電子情報工学科

1　はじめに

Stickel により開発された PTTP [1] は，一階論理コンパイラであり，証明すべき一階論理式が与えられると，その上の Model Elimination[2] を模倣する Prolog プログラムを出力する．出力コードはそれ自身極めて高速に動作するが，トップダウン型の証明を行なう (模倣する) ために，同一解の再計算を頻繁に行なうという本質的な問題を持っている．

本研究では，トップダウン型証明法における再計算の抑制を目的として，中間の計算結果を補題化し，生成された補題を参照することによって証明を高速化する手法を検討する．幾つかの手法を考察し，PTTP に実装，性能評価実験を行なった．

2　補題 (lemma) を用いる方法

トップダウン型証明法において，同一ゴールの再計算といった冗長な探索を避けるために，一度解いた (サブ) ゴールを lemma として保存し，のちの同一ゴールの呼び出しは，この lemma を参照する方法を考える．しかし，素朴にこの機能を組み込んだ場合 lemma を探索することによる探索空間の増大および，lemma 処理のためのオーバヘッドにより，証明の高速化が妨げられる．そこで，本研究ではこれらの問題を解決するため，lemma 処理に以下の制限を付加し，どの制限による処理が有効であるかを実験により検証する．

1. lemma の一般形は節の形をなすが [2]，ここでは，単位節の形になる lemma の生成，利用のみを考える．
2. 独立性試験: 解かれたゴールがテーブルにある lemma に包摂されるときこのゴールはテーブルに保存しない．
3. 一般性試験: 解かれたゴールがテーブルにある lemma より一般的なとき，その lemma を削除する．
4. 登録する lemma の入れ子の深さを制限する．
5. lemma とのマッチングが成功したのち，バックトラッキングが起こってもこの lemma に包摂される入力節の探索は行なわない．

これらの制限だけでは探索空間の増大を抑えることができないため，さらに lemma とのマッチングを制限する方法を考える．通常は，lemma とゴールとの間で単一化を試みるが，ここではゴールが lemma に包摂されるとき，lemma とのマッチングが成功した (lemma にヒットした) とみなす．この方法を用いると，lemma にヒットしてもゴールのバインドは変わらないので，そのゴールについては他の lemma や入力節についての探索をすべてカットすることができる．このため，5を実行する必要もなく，この分の手間を省くこともできる．以上の利点から，今回の実験では，lemma とのマッチングはすべて包摂試験によるもので行なう．

3　実験

a～i のような場合に分けて実験を行なった．その結果を表 1に示す．なお，すべての場合に項の深さ制限を 3 として上の 4を適用した．また，1も，すべての場合について適用する．

a: typeA，独立性試験，indexing，一般性試験

b: typeA，独立性試験，indexing

c: typeA，独立性試験

d: typeB，独立性試験

e: typeA

f: typeB

g: a と同様の処理 + lemma_cost

h: lemma 処理を組み込まない

i: PTTP の結果

typeA,B は，lemma とのマッチングの際，lemma テーブルを参照する方法である．typeA は，lemma テーブルから一つずつ要素を選び出し包摂試験を行なう方法で，typeB は，ゴールを演繹に現れない特殊な定数を用いて例化し，lemma を call するという方法である．indexing は，一般性試験でテー

A useful lemma method for Top-down theorem proving
Yasuomi Chino,Hiroki Ashizawa,Kouji Iwanuma
Yamanashi University
4-3-11 Takeda, Koufu, Yamanashi 400, Japan
E-mail chino@kiritsubo.esi.yamanashi.ac.jp

ブルにあるlemmaを削除する際必要になり，これを用いてlemmaの呼び出しを行なうこともできる．また，gのlemma_costは，ゴールを解くのに要した深さも保存しておき，lemmaがヒットしたときその深さを加算するというものである．通常のlemmaとのマッチングは深さを1だけ消費させる．

4 実験結果

typeBは，項のすべての変数にバインドをかけるため，この項の分解操作を必要とする．このため，扱う項が大きくなる例題には不向きである．e,fは，lemma処理のオーバヘッドがもっとも軽減される場合であるが，lemmaの数が増加し，実行時間が著しく増加した．また，bにおいても生成されるlemmaの数が1.5倍程度に増加した場合は，実行時間もほぼ同程度増加した．オーバヘッドがあるにしても，保存するlemmaの数を抑えるべきであろう．

今回実験したすべての例題において，PTTPより少ないInference回数で解を得ることができた．また，ex4t1, ex4t2, ex5など，PTTPでは停止が確認できなかった例題に対して，解を得ることができた．これらはlemmaとのマッチングを制限したことに起因すると思われる．この制限は，lemmaがヒットしたときには必ず探索空間を狭めることができ，また，決してこれを広げてしまうこともないため，極めて有効な手法であると思われる．

5 今後の課題

lemma処理のオーバヘッドにより，今回の実験でのInference 1回当たりの実行時間はPTTPの6～11倍になる．この遅延をさらに軽減することは依然大きな課題である．実験のhで用いる出力コードは，ほぼPTTPと等しいコードになるが，h，iを比較すると，iの方が格段に効率が良い．これは，PTTPで採用されている効果的な手法，たとえば，フレームリストの分割等を我々のシステムに組み込んでいないためで，これを実装すれば，現在のものよりオーバヘッドが軽減されると予想され，これは現在検討中である．

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.

表 1: 実験結果 (SICStus Prolog2.1, SPARC station IPX)

		a	b	c	d	e	f	g	h	i
ex4t1	Depth	14	14	14	14	14	14	-	-	-
	Time(sec)	89.54	88.391	98.99	199.239	98.93	211.3	-	-	-
	Inference	79942	79942	79942	79942	79942	79942	-	-	-
	NumofLemma	2	2	2	2	18	18	-	-	-
ex4t2	Depth	13	13	13	13	13	13	-	-	-
	Time(sec)	19.291	19.431	21.67	41.83	21.891	42.3	-	-	-
	Inference	22341	22341	22341	22341	22341	22341	-	-	-
	NumofLemma	2	2	2	3	3	3	-	-	-
ex6t2	Depth	11	11	11	11	11	11	14	14	14
	Time(sec)	31.52	33.32	21.19	14.65	908.65	372.99	695.18	44.569	19.17
	Inference	13714	13714	13714	13714	13714	13714	144415	144406	144504
	NumofLemma	24	31	31	31	3124	3124	39	-	-
wos19	Depth	7	7	7	7	7	7	7	7	7
	Time(sec)	17.351	30.58	12.52	11.83	38.6	20.17	24.41	5.04	2.86
	Inference	12962	12962	12962	12962	12962	12962	12970	12970	17687
	NumofLemma	61	98	98	98	1614	1614	61	-	-
ls87	Depth	7	7	7	7	7	7	8	8	6
	Time(sec)	9.701	12.64	6.73	7.66	11.869	10.679	138.44	30.609	3.49
	Inference	8032	8032	8032	8032	8032	8032	59846	60068	16637
	NumofLemma	25	38	38	38	197	197	40	-	-