ナローイングを用いた並行計算のモデルに関する研究*

1M－1

○鈴木泰博†　津本周作‡　田中 博§

東京医科歯科大学難治疾患研究所情報医学研究部門医薬情報¶

1 はじめに

ナローイングは関数型言語と論理型言語の両者の計算メカニズムを包含する計算モデルである。本研究はナローイングを用いて新たな並行計算モデルを提案する。

2 準備

以下で必要な定義を示す。項書換え系、ナローイングの詳細等については例えば[2]を参考にして欲しい。

定義 ナローイング可能 $t|_p \in Oc(t)$ が $\alpha \to \beta \in R$ の左辺と最汎単一化子 θ によって $\alpha\theta = t|_p\theta$ となるときナローイング可能と呼ぶ。

定義 ナローイング $t|_p \in Oc(t)$ が $\alpha \to \beta \in R$ とナローイング可能で、$s = (t[p \leftarrow \beta])\theta$ となるとき t は s にナローイングされるという。

3 並行ナローイング

ナローイングを並行に行うことを考える。例えば、

$$R\begin{cases} f(a) \to c \\ g(b) \to b \end{cases}$$

の規則のもとで、$h(f(x)\ g(x)) =^?\ h(c\ b)$ を並行にナローイングする。すると $h(f(x)\ g(x)) =^?\ h(c\ b) \rightsquigarrow h(c\ b) =^?\ h(c\ b)$ となる。

この場合ナローイングを並行に行うと x への代入が衝突する。Dreshowitz[1] らの提案する抽象並行書換え計算機はこの種の問題を以下の様なアルゴリズムを用い避けている。

ゴールに到達するまで以下の手続きを再帰的に行なう。

1. 項 t のすべての簡約項から並行に簡約を行なう。
2. もし項 s で簡約できなくなったら、s のナローイング可能なすべての部分項を探し単一化子を得る。
3. 単一化子のうち1つを選択しナローイングを行い、再び並行に簡約を行う。解に到達できない場合、全ての単一化子に OR 分岐する。

多くの並行論理型言語も Dreshowitz[1] らの同様の代入方法をとるが、本研究は非同期に変数代入可能な計算モデルを考案した[2]。

4 新たな並行ナローイングについて

定義 構文論的ナローイング[2] $t|_p \in Oc(t)$ が $\alpha \to \beta \in R$ とナローイング可能で、$s = (t[p \leftarrow \beta])$ となるとき、t は s に構文論的ナローイングされるとよぶ。

構文論的ナローイングで共有変数への代入で衝突が起きた場合は後述する Θ の無矛盾チェックを行う。

定義 並行構文論的ナローイング[2] $t|_{p_j} \in Oc(t)(j = 1..k)$ が $\alpha_j \to \beta_j \in R$ とナローイング可能で各 θ_j が無矛盾のとき、t は $s = (t[p_1 \leftarrow \beta_1, \cdots, p_k \leftarrow \beta_k])$ に並行ナローイングされるという。もし無矛盾チェックで矛盾が起きた場合、その単一化子は Θ に入れない。

定義 C ナローイング $t|_{p_j} \in Oc(t)(j = 1..k)$ が $l_j \to_j r \in R$ とナローイング可能ならば、全ての場合に分岐して並行にナローイングを行う。

定義 無矛盾チェック[2] もしある変数 x へ代入が存在し、かつ Θ の環境のもとで異なる $s\theta$ と $t\theta$ が得られるとき、$s\theta$ と $t\theta$ を R で C ベーシックナローイング[2]し $t\theta\downarrow$ と $s\theta\downarrow$ が一致するか否かチェックする。もし $t\theta\downarrow \equiv s\theta\downarrow$ なら無矛盾、$t\theta\downarrow \not\equiv s\theta\downarrow$ なら矛盾とする。

無矛盾チェックで矛盾の場合当該ナローイングは行わない。無矛盾の場合はナローイングを続ける。並行構文論的ナローイングを以下並行ナローイングと呼ぶ。

---

*Study of a concurrent computational model by using narrowing

†Yasuhiro Suzuki

‡Shusaku Tsumoto

§Hiroshi Tanaka

¶Medical Research Institute, Tokyo Medical and Dental University

5 シミュレーション

Cナローイングと、並行ナローイングを並列性の高さ、速度(ナローイングの回数)メモリ量(計算に伴うOR分岐の回数)を性能評価の基準としシミュレーションにより比較した。その結果より、よく性質を表しているものにつき述べる。シミュレーションの結果は表で表す。記号の定義は以下の通り。

or	OR分岐の総数。分岐無しの場合は0。
max	ゴールに至る最大のナローイング回数。
min	ゴールに至る最小のナローイング回数。
ck	無矛盾チェックの総回数
syn narrowing	並行ナローイングを行った場合。

ナローイング回数nは、以下のように数える。式中のtn,cnk,gnは以下の場合のナローイング回数である。

tn ゴールまで。

gn 無矛盾チェックが生じた場所からゴールまで。

ckn 無矛盾チェックにおける。

$$n = \begin{cases} tn & gn \geq cnk \text{ である場合。} \\ tn + (cnk - gn) & gn < cnk \text{ である場合。} \end{cases}$$

シミュレーション1

$$R \begin{cases} g(ab) \to b \\ a \to b \\ g(bb) \to b \end{cases}$$

の規則のもとで、$g(g(ax)g(xb)) =^? b$のシミュレートを行なった。

	or	max	min	ck
C narrowing	8	5	4	
syn narrowing	3	3	2	4

この例では並行性の向上により速度、メモリ量共にCナローイングより向上している。次に

シミュレーション2

$$\mathrm{R} \begin{cases} f(x) \to g(xx) & g(gg) \to c & g(bb) \to c \\ g(ee) \to f(b) & g(dd) \to f(b) & g(ff) \to f(b) \\ f \to h & d \to e & e \to f \\ h \to g \end{cases}$$

のもとで、$f(b) =^? c$のシミュレートを行なった。

	or	max	min	ck
C narrowing	0	2	2	
syn narrowing	12	2	2	22

並行ナローイングは代入を行わない為に規則の探索空間が狭まらずOR分岐増える場合がある。これはその例にあたる。

シミュレーション3

$$R \begin{cases} (x * e) \to x & (l(x * y)) \to ((ly) * (lx)) \\ (l"e") \to e & ((lx) * x) \to e \\ (l(lx)) \to x & (x * ((lx) * y)) \to y \\ ((lx) * (x * y)) \to y & (e * x) \to x \\ ((x * y) * z) \to (x * (y * z)) \end{cases}$$

の規則のもとで、$((x * y) * (x * z)) =^? e$に対して構文論的ナローイングを行なった。以下は2回目のナローイングまでの結果である。

	or	max	min	ck
C narrowing	493	−	−	
syn narrowing	550	−	3	14

並行ナローイングではこのように計算の短縮が生じる場合がある。

6 シミュレーションの結果のまとめ

シミュレーションの結果、並行ナローイングは無矛盾チェック計算が軽いとCナローイングよりOR分岐、最少のナローイング回数が少ない。無矛盾チェックの計算が重いとOR分岐の量が増すが最悪でもCナローイングの最小ナローイング数でゴールに到達する。計算の短縮を行う場合がある。代入をしない為規則の探索空間が絞れずOR分岐が増える場合がある。等がわかった。

7 おわりに

非同期の変数代入が可能な並行計算モデルとして並行ナローイングを提案した。今後は無矛盾チェックが計算に及ぼす影響、計算の短縮等を理論的に検証し、それを踏まえ新たな並行計算言語の設計を行いたい。

参考文献

[1] N.Dershowitz and N.Lindenstrauss, An Abstruct Concurrent Machine for Rewriting, Lecuture Notes in Computer Science 488, pp318-28, 1989.

[2] 鈴木 泰博 酒井 正彦 外山 芳人, 並行ナローイングに関する研究, 北陸先端科学大学院大学 修士論文,1995.