必要平行最外归约策略*

沈 理 孙永强

(上海交通大学计算机系 上海 200030)

摘要 在必要集、必要位置等概念基础上,定义了必要平行最外归约策略.基于最小化必要集思想,该策略适用于正则系统全集,并接近按需调用策略的效率,在适用范围、效率和可实现性三个方面得到了兼顾.

关键词 归约策略,正则重写系统,必要集,必要位置,必要平行最外归约策略.

中图法分类号 TP301

归约策略研究如何正确、高效和方便地将项归约为其范式,是归约系统研究中的重要课题.现有归约策略往往在适用范围、效率和可实现性等方面不能兼顾.目前适用范围最广的是平行最外归约策略[1],其在正则重写系统中是范式化的;许多实际归约系统,如方程式语言,正是基于正则系统的,故该策略的实用性较为广泛.由于在每个归约步同时归约所有最外 redex,平行最外策略含较多的冗余归约,效率不高.为提高效率,一些策略被提出,诸如按需调用策略[2]、最左最外策略[3]和必要集策略[4]等;可惜,在可以实现的前提下,这些策略的适用范围均太窄.

为此,本文定义了必要平行最外归约策略,证明了其在正则系统是范式化的,阐明了其高效性和可实现性,使归约策略在适用范围、效率和可实现性三个方面得到了兼顾.

1 必要集

假定读者熟悉重写系统的基本概念[2],$O(t)$表示项 t 所有出现位置的集合,若 $p\in O(t)$,则 t/p 表示项 t 在位置 p 处的子项,$t[p\leftarrow s]$表示由 s 替代子项 t/p 所得新项.从变量到项的映射叫作替换,项 t 的实例 $\theta(t)$是将替换 θ 作用到 t 中每个变量后所得新项.若 t 为 u 的实例,则称 u 为 t 的前缀,记 $u\leqslant t$ 或 $t\geqslant u$.若 $u\leqslant t$ 且 $u\geqslant t$,则记 $u=t$;若 $u\leqslant t$,且 $u\neq t$,则记 $u<t$或 $t>u$. $fringe(t)=\{p\mid t(p)\in V$,且 $p\in O(t)\}$称作 t 的边缘.若项 t 和 s 拥有一个共同的实例,称其能合一.重写系统的规则左部项被称作模式,模式的实例称之为可归约表达式(Reducible Expression),记为 *redex*.不含 *redex* 的项被称作范式,$nf(t)$表示项 t 的范式.

项 t 关于模式集 L 的最外 *redex* 集合表示为 $orset(t,L)=\{(p,l)\mid l\in L, t/p\geqslant l$,且不存

* 本文研究得到国家自然科学基金、国家"九五"攻关项目和国家优秀奖学金资助.作者**沈理**,1971 年生,博士生,主要研究领域为重写系统与软件工程.**孙永强**,1931 年生,教授,博上导师,主要研究领域为计算机软件.

本文通讯联系人:沈理,上海 200030,上海交通大学计算机系

本文 1996-11-19 收到修改稿

在 $l' \in L, p' < p$,有 $t/p' \geqslant l'$},若 L 明确,简记为 $orset(t)$. 同时,我们也用二元组 (p,l) 表示归约过程 $t \rightarrow_{(p,l)} t'$,即对 t 的位置 p 处 *redex* t/p,应用 l 对应规则实施一次归约,而得到 t' 的过程. 这样,归约序列 $t_0 \rightarrow_{(p_1,l_1)} t_1 \rightarrow_{(p_2,l_2)} t_2 \rightarrow \cdots \rightarrow_{(p_n,l_n)} t_n$ 也可用序列 $(p_1,l_1),(p_2,l_2),\ldots,(p_n,l_n)$ 来表示.

当作用规则 $f(x,y) \rightarrow g(f(x,x))$ 得归约 $f(t_1,t_2) \rightarrow g(f(t_1,t_1))$ 时,t_1 被复制了 2 份,而 t_2 被消除. 直观上,子项 t_1 被繁殖到了右部,故称经过归约 $f(t_1,t_2) \rightarrow g(f(t_1,t_1))$,$t_1$ 在 $g(f(t_1,t_1))$ 中有两个后裔. 后裔的形式定义见文献[2]. 我们称项 t 中的 *redex* s 是必要的,如果在任何从 t 到其范式的归约序列中必有 s 的至少一个后裔被归约. 对于给定重写系统中任何有范式的项 t,若按某策略归约总能得到范式,则称该策略为范式化策略.

按需调用策略每步归约一个必要 *redex*,是强顺序正则系统中的高效策略,但不适用于一般正则系统. 例如,在正则系统 $R=\{B(1,0,x) \rightarrow 1, B(0,x,1) \rightarrow 1, B(x,1,0) \rightarrow 1\}$ 中,$B(R_1,R_2,R_3)$ 中的 3 个 *redex* R_1,R_2,R_3 中的两个是必要的,但无法确定哪两个是必要的. 然而,我们却能发现一个必要集,该集中至少包含一个必要 *redex*,如 $\{R_1,R_3\}$. 必要集策略通过保证每步至少归约一个必要 *redex* 来达到范式化目的,同时提供了最小化冗余归约的可能性.

定义 1.1. 项 t 的 *redex* 集被称为必要集,当且仅当在 t 到其范式的任意归约序列中,至少有一个该集中的 *redex* 或其后裔被归约.

SEKAR[4]证明,每步同时归约当前项的一个必要集中所有 *redex* 的归约策略(称之为必要集归约策略),在正则系统为范式化策略. 但是,SEKAR 仅在构造子正则系统上实施了一种必要集归约策略. 后文将给出一种必要集归约策略,其适用于正则系统全集,不仅效率高,而且易于实现.

2　必要位置

观察必要集归约策略定义不难发现,正则系统中的平行最外策略和强顺序正则系统中的按需调用策略,均属于必要集归约策略. 区别在于,前者保留了冗余归约,后者消除了冗余归约. 由于在正则系统全集中无法完全消除冗余归约,我们的方法在保证范式化基础上,力求将冗余归约尽量减少,以提高归约效率. 为此,引入必要函数集、必要位置等概念来设计尽量避免冗余归约的范式化策略.

由于我们的策略是基于正则系统的,因此下文中的讨论均针对正则系统而言.

定义 2.1. $t \in T(F,V)$ 关于 $p \in O(t)$ 的分支集 $branch(t,p)$ 和最小前缀 $\min prefix(t,p)$ 分别定义为

(1)若 $p=\Lambda$,则 $branch(t,p)=\{\Lambda\}$;否则,$branch(t,p)=O(t) \cap \{p'.i \mid i \in N, p' < p\} \setminus \{p' \mid p' < p\}$.

(2)$\min prefix(t,p)=t[p_1 \leftarrow x_1,\ldots,p_n \leftarrow x_n]$,其中 $branch(t,p)=\{p_1,\ldots,p_n\}$.

$\min prefix(t,p)$ 的直观意义为:保留了项 t 的根到位置 p 路径上的所有符号(不含 p 处符号),并去除了其它符号所得的前缀项. 例如,$\min prefix(f(g(a,b),h(c)),1.2)=f(g(\_,\_),\_)$.

易知如下定理成立,证明见文献[5].

定理 2.2. 设 $t\in T(F,V)$, $p\in O(t)$, $u\equiv \min prefix(t,p)$,有(1)若 $t\leqslant t'$,则 $\min prefix(t',p)\equiv u$;(2)若 $p'\leqslant p$,则 $\min prefix(t/p',p/p')\equiv u/p'$.

由此,可引入必要函数集概念.

定义 2.3. L 为模式集, $t\in T(F,V)$, $p\in O(t)$, $u\equiv \min prefix(t,p)$, t 的 p 位置关于 L 的必要函数集 *nfset* 为

(1)若存在 $p_1,p_2,p_3\in N^*$, $l\in L$,使得 $p=p_1.p_2.p_3$, $u/p_1[p_2\leftarrow x]\leqslant l$,且 $l(p_2)\in V$,则 $nfset(t,p,L)=F\cup V$;

(2)否则, $nfset(t,p,L)=\{l(p/p')\mid p'<p,\ l\in L\cap\{t'\mid t'\geqslant u/p'\}\}$.

必要函数集具有如下性质.

定理 2.4. L 为模式集, $t\in T(F,V)$, $p\in O(t)$. 若 $nfset(t,p,L)\neq F\cup V$,则 $nfset(t,p,L)\cap V=\varnothing$.

证明:若存在 $v\in nfset(t,p,L)\cap V$,可设 $p=p'.p/p'.\wedge$, $l(p/p')\equiv v\in V$, $u\equiv \min prefix(t,p)$, $l\geqslant u/p'\geqslant u/p'[p/p'\leftarrow x]$,故 $nfset(t,p,L)=F\cup V$. 与条件矛盾.

由此可知,必要函数集或者为全集 $F\cup V$,或者为函数集. 例如,重写系统 $R=\{\text{if}(0,x,y)\rightarrow y, \text{if}(1,x,y)\rightarrow x\}$, L 为模式集,设项 $t\equiv \text{if}(\text{if}(0,1,1),\text{if}(1,1,1),\text{if}(0,0,0))$,则 $nfset(t,1,L)=\{0,1\}$, $nfset(t,1.1,L)=\{0,1\}$, $nfset(t,2,L)=F\cup V$. □

定义 2.5. 若 $p\in O(t)$, $t(p)\notin nfset(t,p,L)$,则称 p 为 t 关于 L 的必要位置;否则,称 p 为 t 关于 L 的非必要位置.

上例中,1为 t 的必要位置,1.1和2为 t 的非必要位置. 必要位置具有如下基本性质.

定理 2.6. p 为 t 关于 L 的必要位置,则(1) $nfset(t,p,L)\cap V=\varnothing$;(2)不存在 $p'<p$, $t/p'\geqslant l\in L$,即 p 的祖先节点对应子项无一能被匹配;(3)若 $orset(t/p)\neq\varnothing$,则其为 t 的必要集.

证明:(1)因为 $t(p)\notin nfset(t,p,L)$,所以 $nfset(t,p,L)\neq F\cup V$,由定理2.4可得.

(2)若存在 $p'<p$, $t/p'\geqslant l\in L$,设 $u\equiv \min prefix(t,p)$. 如果 $p/p'\notin O(l)$或 $l(p/p')\in V$,由定义2.3知, $nfset(t,p,L)=F\cup V$, $t(p)\in nfset(t,p,L)$,与定义2.5矛盾,故 $l(p/p')\in F$. 由定理2.2得, $l\geqslant \min prefix(l,p/p')=\min prefix(t/p',p/p')\equiv u/p'$,由定义2.3知 $t(p)\equiv l(p/p')\in nfset(t,p,L)$,矛盾.

(3)只需证,在 t 到其范式的任意归约序列 $Der(t)$ 中,至少有一个 $orset(t/p)$ 中的 *redex* 或其后裔被归约. 设 $len(Der(t))$ 为 $Der(t)$ 的长度,有 $len(Der(t))\geqslant 1$,再设 $Der(t)=(p_1,l_1),Der(t')$,则有 $t\rightarrow(p_1,l_1)t'$. 归纳于 $len(Der(t))$.

a) $len(Der(t))=1$ 时,可得 (p_1,l_1) 为仅有的最外 *redex*,由(2)和 $orset(t/p)\neq\varnothing$ 知, $orset(t/p)=\{(p_1/p,l_1)\}$,命题成立.

b)设 $len(Der(t))<n$ 时命题成立,当 $len(Der(t))=n$ 时,若 $(p_1/p,l_1)\in orset(t/p)$,则命题成立. 否则,可知 (p_1,l_1) 或者是 $orset(t/p)$ 中某个 *redex* 的子项,或者与 $orset(t/p)$ 中的 *redex* 均正交. 对于前者,由于正则系统中内部 *redex* 归约无法消除外部 *redex*,故仍有 p 为 t' 关于 L 的必要位置,且 $orset(t'/p)=orset(t/p)$. 对于后者,亦有 $\min prefix(t',p)=\min prefix(t,p)$,故也有 p 为 t' 关于 L 的必要位置,且 $orset(t'/p)=orset(t/p)$. 因为 $len(Der$

$(t'))<n$,由归纳假定知,$Der(t')$中至少有一个 $orset(t'/p)=orset(t/p)$中的 $redex$ 或其后裔被归约,则命题成立. □

3 必要平行最外归约策略

定理 2.6(3)给出得到必要集的一种方法,找到目标项 t 关于模式集 L 的某个必要位置 p,且有 $orset(t/p)\neq\varnothing$,即得必要集. 一般,任何项的根位置 Λ 总是必要位置,此时 $orset(t)$ 虽为必要集,但仍包含平行最外策略中的冗余归约. 但是,我们可以通过寻找最内必要位置,来最小化相应必要集,尽量避免冗余归约. 为能模块化地求得最内必要位置,引入如下定理.

定理 3.1. p 为 t 关于 L 的必要位置,$p'\in O(t/p)$,则 $nfset(t,p.p',L)=nfset(t/p,p',L)$.

证明:由定义 2.3、定理 2.6(2)直接可得. □

定理 3.2. p_1,p_2 分别为 $t,t/p_1$ 关于 L 的必要位置,则 $p_1.p_2$ 为 t 关于 L 的必要位置.

证明:由定理 3.1 直接可得. □

求得 $nfset(t,p,L)$是判定 p 为 t 关于 L 的必要位置的主要问题,由定义 2.3 知 $nfset(t,p,L)$仅与最小前缀 $u\equiv\min prefix(t,p)$有关,而根据定理 3.1,可模块化地求得 $nfset(t,p,L)$. 亦即,只需给出规模有限的前缀上的必要位置判别方法,再由定理 3.1,通过求得必要位置处子项上各位置的必要函数集,可对规模较大的目标项的必要位置进行判别.

当找到 t 关于 L 的必要位置 p 时,若 $orset(t/p)$非空,则得必要集;否则,引入下述概念,对 $orset(t/p)=\varnothing$情况进行优化处理.

定义 3.3. p 为 t 关于 L 的必要位置,如果 $orset(t/p)=\varnothing$或 $t(p)\notin\{l(\Lambda)\mid l\in L\}$,则称 p 为 t 关于 L 的必要分裂位置;否则,称 p 为 t 关于 L 的必要归约位置.

例 3.4. $L=\{f(a,b,x),g(x,b,c),d\}$,$t\equiv f(g(a,a,c),g(y,h(a,b),d),b)$,则有 $nfset(t,1,L)=\{a\}$,$nfset(t,2,L)=\{b\}$,$nfset(t,2.2,L)=\{b\}$. 因此,1, 2, 2.2 均为必要位置,其中 1, 2.2 为必要分裂位置,2 为必要归约位置.

定义 3.5. t 关于位置 p 的分裂集 $splitset(t,p)$为:$splitset(t,p)=O(t)\cap\{p'.i\mid i\in N,p'<p\}\setminus\{p'\mid p'\leqslant p\}$.

上例中,$splitset(t,2.2)=\{1,2.1,2.2.1,2.2.2,2.3,3\}$. 分裂集满足性质如下.

定理 3.6. 项 t 有范式,p 为 t 关于 L 的必要分裂位置,$splitset(t,p)=\{p_1,\ldots,p_n\}$,则:(1)$t/p_i$ 均有范式,且 $nf(t)\equiv t[p_1\leftarrow nf(t/p_1),\ldots,p_n\leftarrow nf(t/p_n)]$,$1\leqslant i\leqslant n$;(2)任一子项 t/p_i的必要集亦为 t 的必要集,$1\leqslant i\leqslant n$.

证明:设 $u\equiv t[p_1\leftarrow x_1,\ldots,p_n\leftarrow x_n]$,$t\rightarrow t_1\rightarrow t_2\rightarrow\ldots\rightarrow t_n\equiv nf(t)$为任一 t 的范式化序列. 由定义 3.3 和归纳法易证,对于 $1\leqslant i\leqslant n$,有 $u\leqslant t_i$,且 p 为 t_i 关于 L 的必要分裂位置. 亦即,归约操作始终只在子项 t/p_i 中进行,又 t/p_i 彼此两两正交,其归约互不影响,显然可得(1)、(2)成立. □

这样,若项 t 中有必要分裂位置 p,则可用逐个求得分裂集 $splitset(t,p)$对应各子项范式的方法来达到范式化目的. 这种分裂归约有利于我们更早地缩小考虑问题的范围,在模式匹配和选取必要集方面,都有提高效率的可能性,是一种有效的加速归约技术.

基于必要位置和必要分裂位置具有的性质,下面来定义必要平行最外归约策略.

算法 3.7. 函数 $Nmlform(t)$返回 $nf(t)$,函数 $nfstep(t)$对 t 实施一定归约,返回归约结果.

```
Function Nmlform(t)
1 while orset(t)≠∅ do
2       t=nfstep(t)
3 return t
Function nfstep(t)
1 while 存在 p=Λ为t关于L的必要分裂位置或 p≠Λ为必要位置 do
2 if p 为必要分裂位置 then
3       for 每个 p'∈splitset(t,p) do
4             t=t[p'←Nmlform(t/p')]
5             return t
6       else
7             repeat
8                 t=t[p←nfstep(t/p)]
9             until p 为非必要位置或必要分裂位置
10 if orset≠∅ then
11       for 每个(p',l)∈orset(t) do
12             t=t[p'←replace(t/p',l)]
               /* replace(t,l)实施归约 t→(p,l)t',返回 t' */
13 return t
```

我们称算法 3.7 所定义的策略为必要平行最外归约策略(Needed Parallel-Outermost Reduction Strategy),简称 NPO 归约策略.为便于理解,下面用非形式化方法描述该策略.t 为目标项,L 为模式集,则归约方式如下:

(1) 若 t 中有必要分裂位置或非根必要归约位置 p,转(2);否则,转(3).

(2) 若 p 为必要分裂位置,则用 NPO 策略逐个范式化 $splitset(t,p)$对应子项,结束;否则,用 NPO 策略逐步归约子项 t/p,至 p 为非必要位置或必要分裂位置止,转(1).

(3) 若 $orset(t)\neq\varnothing$,则归约之,转(1);否则,结束.

下节我们将阐明 NPO 策略是一种高效、易实现的必要集归约策略,是适用于正则系统全集的理想归约策略.

4 NPO 策略性质

考虑到 NPO 策略的归约操作仅在算法 3.7 的 $nfstep$ 函数的 11、12 行语句中实施,为描述方便,引入初始位置概念.

定义 4.1. t 为目标项,$p\in O(t)$.若由于调用 $nfstep(t)$而生成的所有被调用的 $nfstep$ 函数中,$nfstep(t/p)$第 1 个执行 11、12 行语句,则称 p 为 t 的初始位置,t/p 为 t 的初始子项.

定理 4.2. t 为目标项,$orset(t)\neq\varnothing$,则 t 有初始位置 p.

证明:归纳于 t 所含符号数 $size(t)$.

(1)$size(t)=1$ 时,$orset(t)=\{(\Lambda,l)\}$,$nfstep(t)$本身执行 11、12 行,Λ 为 t 的初始位置.

(2)设 $size(t)<n$ 时命题成立,当 $size(t)=n$ 时,分 3 种情况:

a) $nfstep(t)$前 3 步执行语句 1~3.设相应必要分裂位置为 p,3、4 句依次对 $splitset(t,p)$中的子项 $t/p_1,t/p_2,\ldots,t/p_n$ 实施范式化.因为 $orset(t)\neq\varnothing$,则由定理 3.6 知,存在 $1\leqslant i$

$\leqslant n$,$orset(t/p_i)\neq\varnothing$,且 $orset(t/p_j)=\varnothing$,$1\leqslant j\leqslant i$. 显然,$Normalform(t/p_j)$均可终止,故可由 $Normalform(t/p_i)$执行至 $nfstep(t/p_i)$. 而 $size(t/p_i)<n$,由归纳假定,t/p_i 有初始位置 p',则 $p_i.p'$ 为 t 的初始位置.

b) $nfstep(t)$前3步执行语句1,2,7. 设相应必要归约位置为 $p\neq\Lambda$,有 $size(t/p)<n$,由归纳假定,t/p 有初始位置 p',由第8句 $nfstep(t/p)$知 $p.p'$ 为 t 的初始位置.

c) 除a),b)之外,则 $nfstep(t)$可执行至11、12行,Λ 有为 t 的初始位置. □

为证明NPO策略为必要集策略,我们引入必要集位置概念.

定义 4.3. t 为项,L 为模式集. (1)若 p 为 t 关于 L 的必要位置,且 $orset(t/p)\neq\varnothing$,则 p 为 t 关于 L 的必要集位置. (2)若 p_1 为 t 关于 L 的必要分裂位置,$p_2\in splitset(t,p_1)$,p_3 为 t/p_2 关于 L 的必要集位置,则 $p_2.p_3$ 为 t 关于 L 的必要集位置.

定理 4.4. p 为 t 关于 L 的必要集位置,则 $orset(t/p)$ 为 t 的必要集.

证明:施归纳于 $size(t)$.

(1)$size(t)=1$ 时,$O(t)=\{\Lambda\}$. 若 Λ 为必要集位置,则 $orset(t/\Lambda)$ 显然为必要集.

(2)设 $size(t)<n$ 时命题成立,当 $size(t)=n$ 时,分两种情况:

a) 存在若 p_1 为 t 关于 L 的必要分裂位置,$p_2\in splitset(t,p_1)$,p_3 为 t/p_2 关于 L 的必要集位置,有 $p=p_2.p_3$. 因为 $size(t/p_2)<size(t)=n$,由归纳假定知 $orset(t/p)$ 为 t/p_2 的必要集,由定理3.6知其为 t 的必要集.

b) 否则,p 为 t 关于 L 的必要位置,由定理2.6(3)知命题成立. □

由此可证NPO策略为必要集策略.

定理 4.5. NPO策略为必要集策略.

证明:分析算法3.7可知,真正的归约操作是对当前目标项 t 的某个子项 t/p,由 $nfstep(t/p)$调用第11、12行语句来实施的. 由定义4.3和定理3.2可知,此时的 p 为 t 的必要集位置;再由定理4.4知,第11、12行归约的 $orset(t/p)$正是 t 的一个必要集. 可见,NPO策略归约的始终是当前目标项 t 的必要集. 再设 t 为NPO策略归约过程中任一当前目标项,且不为范式. 若 $nfstep$ 函数执行过程中无法实施归约,则必能返回 $Normalform(t)$,并调用 $nfstep(t)$,由定理4.2知存在初始位置 p,即能找到 t 的必要集 $orset(t/p)$实施归约. 换言之,对于每个归约中的非范式项 t,NPO策略均能找到 t 的一个必要集实施归约,至其为范式止. 所以,命题得证. □

作为必要集策略,NPO策略在正则系统中是范式化的. 通过不断寻找更内部的必要位置,并尽可能实施分裂归约,NPO策略努力求得最内的必要集位置,来保证得到的必要集尽可能小,避免冗余归约,提高归约效率.

例如,对于项 $t\equiv$if(if(0,1,0),if(1,0,1),if(1,1,0)),采用平行最外策略,有归约序列 $t\rightarrow_{1,2,3}$if(0,0,1)$\rightarrow_\Lambda 1$;而采用NPO策略,有归约序列 $t\rightarrow_1$if(0,if(1,0,1),if(1,1,0))$\rightarrow_\Lambda$if(1,1,0)$\rightarrow_\Lambda 1$. 显然,后者消除了冗余 $redex\ t/2\equiv$if(1,0,1)的归约,最小化了必要集,提高了效率.

采用分裂归约还可带来两点好处:(1)缩小目标项的考虑范围,以此降低归约过程中模式匹配的开销;(2)由下例可见,对子项逐个进行范式化还有利于降低空间开销.

例 4.6. 重写系统 $R=\{ad(0,x)\rightarrow x, ad(s(x),y)\rightarrow s(ad(x,y)), db(x)\rightarrow ad(x,x)\}$和项

$t \equiv f(db(s(0)), db(s(0)))$. 若用平行最外策略，有归约序列 $t \to f(ad(s(0), s(0)), ad(s(0), s(0))) \to f(s(ad(0, s(0))), s(ad(0, s(0)))) \to f(s(s(0)), s(s(0)))$，此序列中最大项规模为11. 而用NPO策略，归约序列为 $t \to f(ad(s(0), s(0)), db(s(0))) \to f(s(ad(0, s(0))), db(s(0))) \to f(s(s(0)), db(s(0))) \to f(s(s(0)), ad(s(0), s(0))) \to f(s(s(0)), s(ad(0, s(0)))) \to f(s(s(0)), s(s(0)))$，此序列中最大项规模为9. 由于最大项规模可作为空间开销的衡量指标，故分裂归约有利于降低空间开销.

可见，NPO策略既是正则系统中的范式化策略，又是一种时空效率颇高的归约策略，另外，若我们采用自上而下的模式匹配方式时，必要位置和必要分裂位置的判定易于实现，因此NPO策略的实现也是不难的.

5 实现简况

应用文献[5,6]中的模式匹配方法和图归约技术，我们分别实现了贯彻平行最外策略和NPO策略的规范化系统，对给定具体重写系统(定义详见文献[5])中项的范式化情况参见表1，可见后者在时间和空间上都取得了很好的效果.

表1

目标项	归约步数	执行时间(*ticks*)	所需最多项节点数
$nfold(reverse(y), list(s(s(s(s(s(s(o)))))))), s(s(s(o))))$	240:178	34:14	330:242
$fib(s(s(s(s(s(s(s(s(s(o)))))))))$	293:293	28:25	189:95
$merge(list(s(s(s(o)))), list(s(s(o))))$	89:55	15:7	284:206

单元格中“:”左部值针对平行最外策略，“:”右部值针对NPO策略.

表2

归约策略	适用范围	效率	可实现性
NPO策略	正则系统全集	高	好
最左最外策略	正则系统子集—函数式语言	高	好
按需调用策略	正则系统子集—强顺序系统	高	好
平行最外策略	正则系统全集	低	好
Sekar的必要集策略	理论上适用于正则系统全集，但具体实现方法仅适用于构造子正则系统	可高可低，由具体实现而定	差

NPO策略不仅克服了平行最外策略的低效性，还拥有了当前一些著名归约策略所难以兼顾的多项优点. 表2对这些归约策略在适用范围、效率和可实现性3方面进行了比较，从中分析可得：基于最小化必要集思想，NPO策略是正则系统中一种高效、易实现的范式化策略，使归约策略在适用范围、效率和可实现性3方面得以兼顾.

对于NPO策略在非正则系统中的适用情况以及如何将之应用于实际系统中，如文献[7]中的基于重写方法的程序开发系统，我们将进一步深入研究.

参考文献

1 O'Donnell M J. Equational logic as a programming language. MIT Press, 1985.

2 Klop J W, Middeldrop A. Sequentiality in orthogonal term rewriting systems. JSC, 1991, **12**:161～195.

3 Peyton Jones, S. L. The implementation of functional programming languages. Prentice Hall, 1987.

4 Sekar R C, Ramakrishnan I V. Programming in equational logic: beyond strong sequentiality. Proc. IEEE Symp.

on Logic in Computer Science, 1990. 230～241.
5 沈理. 重写系统中快速规范化技术的研究与实现[硕士论文]. 上海交通大学, 1996.
6 沈理, 林凯, 孙永强. 平行最外模式匹配. 软件学报, 1996, 7(增刊): 329～337.
7 林凯, 孙永强. 基于重写方法的程序开发系统的设计和实现. 计算机学报, 1996, 19(9): 641～648.

NEEDED PARALLEL-OUTERMOST REDUCTION STRATEGY

SHEN Li　SUN Yongqiang

(*Department of Computer Science　Shanghai Jiaotong University　Shanghai*　200030)

Abstract　Based on the concepts such as needed set, needed position and so on, needed parallel-outermost reduction strategy is introduced in this paper. By minimizing needed sets, this strategy is complete for the regular term rewriting systems, is efficient like call by need strategy, and can satisfy all demands of applicability, efficiency and computability.

Key words　Reduction strategy, regular term rewriting system, needed set, needed position, needed parallel-outermost reduction strategy.

Class number　TP301