

Capítulo 9

Sistemas de reescrita condicionais (CTRSs)

Os sistemas de reescrita condicionais (*conditional term rewriting systems* (CTRs)) aparecem no contexto algébrico com o objetivo de operacionalizar a dedução em teorias apresentadas condicionalmente. Estudam-se brevemente as propriedades da reescrita condicional e suas aplicações computacionais.

9.1 Introdução

Os **sistemas de reescrita condicionais**, por brevidade **CTRSs**, representam a mais importante extensão dos sistemas de reescrita de termos. Tal extensão admite regras de reescrita com condições equacionais. A inclusão de condições gerais dentro das especificações algébricas puramente equacionais foi proposta por O'Donnell na década de 70 [O'D77], mas somente no princípio dos 80 aparecem os primeiros estudos profundos para o caso de condições puramente equacionais realizados por Kaplan [Kap83], [Kap84a]. Kaplan estuda os diferentes mecanismos de avaliação das condições e seleciona a avaliação padrão demonstrando a correção desta pelo teorema do ponto fixo. Para atacar o problema da indecidibilidade da reescrita dos CTRSs padrão, Kaplan define a propriedade de simplificação [Kap84b] e Jouannaud e Walmann definem os CTRSs reductivos [JW86]; tal propriedade (e classe de CTRSs) são logo generalizados por Sivakumar na bem conhecida propriedade de *decrecimento* [Siv89] (veja também [DOS87], [DOS88]). Outros trabalhos relevantes no tratamento dos CTRSs são o trabalho de Remy e Zhang [ZR85] e a tese de Navarro [NG87], onde se define um certo tipo de reescrita hierárquica, que posterga a avaliação das condições. Adicionalmente, um dos autores trata um tipo especial de sistemas condicionais que incluem, além das condições equacionais, condições gerais em teorias de primeira-ordem com certas restrições, desde que seja possível misturar a reescrita pura com outros mecanismos computacionais (como a programação lógica) que muitas vezes resultam mais eficientes [AR95a, AR95b, dA98, AR00].

As propriedades de confluência e terminação para os CTRSs são tratadas em profundi-

dade por Bergstra e Klop [BK86] e por Gramlich [Gra95]. Um dos autores também pesquisou estas propriedades para os CTRSs com condições gerais [AR94, dA98, AR00]. Pode também ser consultado o livro [Ohl02].

Entre outros, Jouannaud e Walmann [JW86], Kaplan e Remy [KR89], Sivakumar [Siv89], Ganzinger [Gan88, Gan91] e Ohlebusch [Ohl99] têm tratado a completção dos CTRSs.

9.2 Equações Condicionais

Os sistemas de reescrita condicionais aparecem originalmente no contexto da Álgebra Universal e na Teoria de Tipos Abstratos de Dados como implementações de especificações com axiomas equacionais positivos da forma:

$$u_1 = v_1 \wedge \dots \wedge u_n = v_n \Rightarrow l = r$$

onde u_i, v_i, l, r são termos em $Term(F, V)$ como no caso não condicional. Se $n = 0$ a equação é não-condicional. As *premissas* ou *condição*, $u_1 = v_1 \wedge \dots \wedge u_n = v_n$, da equação são frequentemente denotadas por c e a *conclusão*, $l = r$ é denominada *conclusão*. Tal tipo de equações é denominado cláusulas equacionais de Horn.

Definição 9.2.1 *Um sistema de equações condicionais, E , é um conjunto finito de equações condicionais.*

As equações condicionais são implicitamente quantificadas universalmente. Consequentemente as *variáveis extras*, i.e., aquelas que ocorrem na premissa, mas não na conclusão de uma equação condicional, têm semântica existencial. Por exemplo, a equação condicional:

$$\forall x, y, z \{ \{x = y \wedge y = z\} \Rightarrow x = z \}$$

é equivalente a

$$\forall x, z \{ \exists y \{x = y \wedge y = z\} \Rightarrow x = z \}$$

Exemplo 9.2.2 O *mdc* (*maior divisor comum*) de dois números naturais pode ser especificado com equações condicionais com 0 e sucessor, s , como segue:

$$\begin{array}{l} : \quad 0 < 0 = 0 \\ : \quad 0 < s(0) = s(0) \\ : \quad s(x) < 0 = 0 \\ : \quad s(x) < s(y) = x < y \\ : \quad s(x) - s(y) = x - y \\ : \quad 0 - x = 0 \\ : \quad x - 0 = x \\ : \quad mdc(x, x) = x \\ y < x = s(0) : mdc(x, y) = mdc(x - y, y) \\ x < y = s(0) : mdc(x, y) = mdc(x, y - x) \end{array}$$

Para manter a especificação mono-sortida, 0 e $s(0)$ são usados como constantes booleanas *false* e *true*, respectivamente. Adicionalmente, “ $-$ ” é a subtração truncada.

◇

As especificações condicionais não somente são mais elegantes que as puramente equacionais. Elas são mais expressivas. Existem muitos exemplos na literatura das especificações condicionais que não podem ser apresentadas por meio de axiomas puramente equacionais. Também tem-se tratado transformar sintaticamente especificações condicionais em não condicionais. Geralmente tais mecanismos sintáticos resultam apropriados sob fortes restrições. Veja por exemplo [Hin95b, Hin95a], [AR96], [Ohl99].

Exemplo 9.2.3 Considere o seguinte sistema de equações condicionais:

$$\begin{array}{l} : f(g(x)) = c \\ : f(h(g(x))) = c \\ f(h(x)) = f(x) : f(h(h(x))) = f(h(x)) \end{array}$$

A condição $f(h(h(x))) = f(h(x))$ é decidível em tal sistema, i.e., $f(h(y)) = f(y)$ é aplicável se e somente se $y \in \{h^n(g(x)) | n \geq 0\}$. É simples construir um sistema puramente equacional equivalente, mas infinito:

$$\begin{array}{l} f(g(x)) = c \\ f(h(g(x))) = c \\ \vdots \\ f(h^n(g(x))) = c \\ \vdots \end{array}$$

Para demonstrar que não existe um sistema puramente equacional equivalente e finito sem aumentar a assinatura (i.e., sem *funções escondidas*), suponha que exista um tal sistema com equações $E = \{s_1 = t_1, \dots, s_n = t_n\}$. Considere, então, um termo da forma $f(h^k(g(x)))$ “maior” que cada lado $s_i, t_i, 1 \leq i \leq n$ das equações, onde “maior” significa com mais ocorrências de símbolos f, g, h . A suposição implica que de $f(h^k(g(x)))$ se deduz equacionalmente c , i.e., $f(h^k(g(x))) =_E c$, mas a classe de E -equivalência de $f(h^k(g(x)))$ deve ser unitária, já que qualquer subtermo próprio de $f(h^k(g(x)))$ tem classe de equivalência unitária.

◇

Em analogia com o caso puramente equacional, uma semântica inicial para o caso condicional pode ser desenvolvida. Da mesma maneira que para especificações equacionais, existe um sistema dedutivo com seu teorema de completude correspondente (similar ao de Birkhoff para dedução equacional).

Definição 9.2.4 *Seja (SIG, E) um sistema de equações condicionais sobre uma assinatura SIG e seja $u_1 = v_1 \wedge \dots \wedge u_n = v_n \Rightarrow l = r$ qualquer equação condicional em E . Um modelo de E é uma SIG -álgebra \mathcal{A} tal que para qualquer instância $\phi : V \rightarrow \mathcal{A}$ se $\forall i, 1 \leq i \leq n, (u_i\phi)^{\mathcal{A}} = (v_i\phi)^{\mathcal{A}}$ então $(l\phi)^{\mathcal{A}} = (r\phi)^{\mathcal{A}}$.*

Isto é consistente com a noção usual de modelo para teorias axiomatizadas equacionalmente. Denota-se com Alg_E a classe de todos os modelos de E . A noção de consequência lógica é estendida de uma maneira direta.

Definição 9.2.5 *Uma equação condicional $c \Rightarrow l = r$ é uma consequência lógica de um sistema de equações condicionais (SIG, E) , denotado por $E \models (c \Rightarrow l = r)$, se $c \Rightarrow s = t$ é válida em todos os modelos de E .*

O sistema na Tabela 9.1 é um conjunto de regras de inferência para a lógica equacional condicional:

$\Rightarrow t = t$	reflexividade
$s = t \wedge s' = t \Rightarrow s = t$	consq. trivial
$t_1 = t_2 \wedge t_2 = t_3 \Rightarrow t_1 = t_3$	transitividade
$s_1 = t_1 \wedge \dots \wedge s_n = t_n \Rightarrow f(t_1, \dots, t_n) = f(s_1, \dots, s_n)$	preservação
$\frac{\Rightarrow t_1 = t_2}{\Rightarrow t_2 = t_1}$	simetria
$\frac{E \wedge u = v \Rightarrow l = r, F \Rightarrow u = v}{E \wedge F \Rightarrow l = r}$	premissa válida
$\frac{E \Rightarrow l = r}{E\sigma \Rightarrow l\sigma = r\sigma}$	substituição

Tabela 9.1: Regras de inferência para a lógica equacional condicional

Onde $E = \{t_1 = s_1 \wedge \dots \wedge t_n = s_n\}$ e $F = \{t'_1 = s'_1 \wedge \dots \wedge t'_m = s'_m\}$ são premissas ou condições, e σ uma substituição.

Quando uma equação condicional $c \Rightarrow s = t$ é derivável por meio deste sistema de regras de inferência, usando as equações condicionais de E como axiomas, se escreve $E \vdash (c \Rightarrow s = t)$.

Finalmente tem-se uma extensão do teorma de Birkhoff.

Teorema 9.2.6 (Teorema de Birkhoff para especificações condicionais) *Seja (SIG, E) uma especificação equacional condicional. Então*

$$E \models u_1 = v_1 \wedge \dots \wedge u_n = v_n \Rightarrow s = t \text{ sse } E \vdash u_1 = v_1 \wedge \dots \wedge u_n = v_n \Rightarrow s = t$$

Para um tratamento mais formal veja por exemplo [Kap84a] e para um tratamento preciso introdutório que discrimina teorias equacionais condicionais e puramente equacionais no contexto da álgebra universal veja [Wec92], ou o capítulo 2 de [AR93].

9.3 Semântica operacional dos sistemas condicionais equacionais

Um sistema condicional **equacional** R , é um conjunto de cláusulas de Horn da forma:

$$s_1 = t_1 \wedge \dots \wedge s_n = t_n : l = r.$$

$s_1 = t_1 \wedge \dots \wedge s_n = t_n$ são as premissas ou condições da equação condicional e $l = r$, a sua conclusão. l e r são denominados os lados esquerdo e direito da equação, respectivamente. Define-se a relação de simplificação em um passo \sim e o seu fecho reflexivo-transitivo \sim^* como: Se $s_1 = t_1 \wedge \dots \wedge s_n = t_n : l = r$ é uma equação condicional em R , σ é uma substituição, u é um termo, π é uma posição em u e $t_i\sigma \sim^* s_i\sigma$ para $1 \leq i \leq n$, então $u[l\sigma]_\pi \sim u[r\sigma]_\pi$. Se $s \sim^* t$, escreve-se $R \vdash s = t$, ou simplesmente $s = t$.

Um CTRS **natural** R tem regras da forma:

$$s_1 \leftrightarrow^* t_1 \wedge \dots \wedge s_n \leftrightarrow^* t_n : l \rightarrow r.$$

Quando uma tal regra se aplica, uma instância $l\sigma$ de l num termo s é substituída por $r\sigma$, gerando um termo t . Escreve-se $R \vdash s \rightarrow t$, ou simplesmente $s \rightarrow t$. A regra pode ser aplicada somente se existe uma prova $s_i \leftrightarrow^* t_i$ para cada uma das suas condições instanciadas correspondentemente (utilizando a substituição σ). Se $n = 0$ a regra é dita não condicional.

Um sistema de reescrita natural não é muito diferente do sistema subjacente equacional. Cada regra da forma acima corresponde à equação condicional obtida substituindo “ \rightarrow ” por “ $=$ ”, “ \leftrightarrow^* ” por “ $=$ ” e “ $:$ ” por “ \supset ”. Denotar-se-á como R^{eqn} o sistema equacional subjacente obtido do sistema natural R^{nat} desta forma. Da mesma forma R^{nat} denotará o sistema natural associado a um sistema equacional R^{eqn} .

Um resultado apresentado em [DO88] estabelece a equivalência entre CTRS naturais e sistemas equacionais como a seguir.

Para qualquer CTRS natural R^{nat} , $R^{nat} \vdash p \leftrightarrow^* q$ se, e somente se $R^{eqn} \vdash p = q$

Uma prova $p \leftrightarrow^* q$ num sistema natural é conseqüentemente denominada uma **prova equacional**.

Da teoria de reescrita clássica, sabemos que a **propriedade de confluência** ($p \downarrow q$ sempre que $u \rightarrow^* p$ e $u \rightarrow^* q$ para algum u) é equivalente à **propriedade de Church-Rosser** (qualquer prova equacional $p \leftrightarrow^* q$ pode ser substituída por uma **prova de reescrita normal**: $p \downarrow q$). Assim, para qualquer sistema confluyente, e forma normal t , $p \leftrightarrow^* t$ implica $p \rightarrow^* t$. Desta forma, reescrita com sistemas confluyentes pode ser utilizada para achar formas normais.

Como consequência direta da equivalência entre CTRS naturais e sistemas equacionais, tem-se que para todo CTRS natural confluyente R^{nat} e sistema condicional equacional subjacente R^{eqn} , provas normais em R^{nat} são equivalentes a provas equacionais em R^{eqn} (i. e. $R^{nat} \vdash p \downarrow q$ se, e somente se $R^{eqn} \vdash p = q$).

Um problema evidente com CTRS naturais é que as condições para poder aplicar uma regra envolvem demonstrações arbitrárias de igualdades. Assim, tem-se ganhado muito pouco da noção de reescrita. Para solucionar essa limitação, consideram-se definições mais restritas de CTRS:

Um CTRS **padrão (ou *join*)** [Kap84b] é um conjunto de regras da forma:

$$s_1 \downarrow t_1 \wedge \dots \wedge s_n \downarrow t_n : l \rightarrow r,$$

significando que uma instância $l\sigma$ de l reduz para $r\sigma$ se cada par na condição $s_i\sigma$ e $t_i\sigma$ pode ser reduzido para um mesmo termo, $i = 1 \dots n$. Para um CTRS padrão R^{std} , R^{eqn} e R^{nat} denotarão o sistema equacional e natural associados, respectivamente. Da mesma forma R^{std} será utilizado notacionalmente.

Um resultado em [DO88] estabelece a equivalência entre a propriedade de juntabilidade em CTRS padrão e a igualdade em sistemas equacionais, como apresentado a seguir.

Para qualquer CTRS padrão confluyente R^{std} , $R^{std} \vdash p \downarrow q$ se, e somente se $R^{eqn} \vdash p = q$.

É fácil verificar que se R^{std} é confluyente, então R^{eqn} é confluyente, assim o é o sistema natural R^{nat} (como consequência dos teoremas precedentes tem-se $R^{nat} \vdash p \leftrightarrow^* q$ implica $R^{std} \vdash p \downarrow q$, que é uma prova em R^{nat}), mas o outro sentido não vale, já que as condições numa prova normal não necessariamente são transformáveis em provas normais. Precisam-se propriedades adicionais para obter provas normais em sistemas padrão daquelas em sistemas naturais.

CTRSs padrão correspondem à intuição sobre reescrita condicional. Note que CTRS padrão também permitem avaliação recursiva das premissas.

9.4 Complexidade das formas normais

Apesar de que no caso não condicional, estar em “forma normal” é uma propriedade (facilmente) decidível, isso não é mais certo no caso condicional. Com efeito, existem CTRS padrão para os quais o conjunto de formas normais é indecidível.

Teorema 9.4.1 (Indecidibilidade da aplicabilidade das regras condicionais) *Existe um sistema condicional com uma única regra R sobre uma assinatura dada SIG , tal que o problema de decidir a redução (aplicabilidade da regra) é indecidível, e a função de forma normal associada é não computável.*

Demonstração. (Idéia geral [Kap84a]). Usa-se a resposta de Matjacevič's ao décimo problema de Hilbert, que diz que é indecidível a questão de que se um polinômio inteiro tem ou não raiz inteira.

Pode-se definir um tipo abstrato, sobre uma assinatura adequada e regras equacionais, tais que o modelo inicial \mathcal{I} é isomorfo ao anel $\mathbb{Z}[x]$ de polinômios inteiros. Por exemplo, o polinômio $x^3 - 2x + 1$ será representado pelo termo:

$$x \uparrow (s(s(s(0)))) + (p(p(0))) * x \uparrow (s(0)) + x \uparrow 0$$

com:

$$\begin{aligned} x \uparrow &: \text{int} && \rightarrow \text{polyn} \\ * &: \text{int polyn} && \rightarrow \text{polyn} \\ + &: \text{polyn polyn} && \rightarrow \text{polyn} \end{aligned}$$

Adiciona-se então o seguinte predicado:

$$\text{Root} : \text{polyn int} \rightarrow \text{bool},$$

com a regra

$$eq(p[abs(n)], 0) \text{ ou } eq(p[-abs(n)], 0) \text{ ou } \text{Root}(p, abs(n) + 1) = true : \text{Root}(p, abs(n)) \rightarrow true$$

O operador $_[-]$: $\text{polyn int} \rightarrow \text{int}$ é suposto ser de tipos para simular a aplicação de polinômios sobre inteiros. Os operadores abs , $-$, eq , or e constantes 0 e $true$ têm a interpretação de intenção. Com essa regra de reescrita condicional, pode-se provar que o termo $\text{Root}(p, abs(n))$ reescreve em $true$ se, e somente se P tem uma raiz de módulo maior ou igual a n . Caso contrário é irreduzível.

A interpretação dos axiomas é: P tem raiz de valor absoluto maior ou igual a um inteiro positivo n se:

- $P[n] = 0$ ou
- $P[-n] = 0$ ou
- P tem uma raiz de valor absoluto maior ou igual a $n + 1$.

Assim, é indecidível se o termo $Root(p, 0)$ (para qualquer p) é reductível \rightarrow_R , já que isso significaria que se poderia decidir se (qualquer) p tem ou não raiz.

As formas normais via \rightarrow_R de aqueles termos não são computáveis, pelo mesmo motivo. □

Esse é um dos principais problemas da teoria: a própria aplicabilidade das regras de reescrita condicionais é indecidível. O ponto é evitar chamados recursivos infinitos produzidos pela decidibilidade das premissas. Nessa direção têm sido utilizados conceitos como CTRS *reduativos* [JW86] ou *decrementais* [DOS88]. Ver-se-á o último.

Note também que esse problema aparece de forma simples operacionalmente. Considere por exemplo o sistema condicional da seguinte regra.

$$f(x) = f(f(x)) : f(x) \rightarrow a$$

Para reduzir $f(a)$ deve-se verificar (operacionalmente) se $f(f(a))$ reduz para $f(a)$ e assim recursiva e subsequentemente.

Definição 9.4.2 (CTRSs Decrementais) *Um sistema padrão e natural é denominado **decremental** se existe uma extensão bem-fundada \succ da relação de reescrita \rightarrow que satisfaz adicionalmente as seguintes propriedades:*

- \succ contém a relação de subtermos próprios (i.e., se s é um subtermo próprio de t , então $t \succ s$);
- para cada regra $s_1 \bullet t_1 \wedge \dots \wedge s_n \bullet t_n : l \rightarrow r$, $l\sigma \succ s_i\sigma, t_i\sigma$, para cada substituição σ e índices $1 \leq i \leq n$. Onde \bullet representa \leftrightarrow^* e \downarrow para sistemas naturais e padrão, respectivamente.

Diz-se que uma prova $s \leftrightarrow^* t$ é **completamente normal** se é uma prova normal $s \downarrow t$ e existem sub-provas completamente normais das condições $c_i \leftrightarrow^* d_i$ usadas na prova de $s \downarrow t$.

Note que qualquer prova normal em R^{std} é também uma prova normal em R^{nat} . No outro sentido, em [DO88] foi demonstrado que para sistemas condicionais naturais decrementais confluentes qualquer prova normal pode ser transformada numa prova completamente normal, que vale no sistema padrão subjacente. Esse resultado pode ser resumido da seguinte maneira:

Para qualquer CTRS natural, decremental e confluyente, $R^{nat} \vdash p \downarrow q$ sse $R^{std} \vdash p \downarrow q$.

A seguir ir-se-ão considerar unicamente CTRS padrão e ir-se-á escrever simplesmente $u_i = v_i$ (mas não $u_i \downarrow v_i$) para as premissas nas condições padrão.

Teorema 9.4.3 (Decidibilidade para CTRSs Padrão Decrementais) *Seja R um CTRS padrão decremental. As noções básicas são decidíveis; i.e., a relação de reescrita \rightarrow_R , a relação de derivabilidade \rightarrow^* , a relação de juntabilidade \downarrow , e o atributo de ser forma normal, são todas recursivas.*

Demonstração. Primeiro observe que o sistema é obviamente terminante como consequência da propriedade de que \succ é bem-fundada. A decidibilidade das noções básicas é provada por indução transfinita sobre \succ , como a seguir. Primeiro, considerar-se-á a seguinte propriedade: “Dado um termo t pode-se encontrar um conjunto de formas normais de t ”. Se t não tem subtermos instância de lados esquerdos das regras do sistema, então t é irreduzível sendo a sua própria forma normal. Caso contrário, seja $t \equiv u[l\sigma]$ para alguma regra $u_1 = v_1 \wedge \dots \wedge u_n = v_n : l \rightarrow r$. Pelas duas condições da definição de decremental, tem-se que $t \equiv u[l\sigma] \succ l\sigma$ e $l\sigma \succ u_i\sigma, v_i\sigma$. Por indução, já que $t \succ u_i, v_i$, o conjunto de formas normais de u_i, v_i , para cada i , pode ser computado e dessa forma verificado se a regra aplica. Nesse caso, $t \rightarrow u[r\sigma]$. Similarmente, (utilizando cada regra que casa) podem ser computados todos os termos, sejam estes s_i, \dots, s_k para os quais t reduz num passo. Pela hipótese de indução, as formas normais de cada s_i podem ser enumeradas. Então a união desses termos é o conjunto de formas normais de t . As outras propriedades básicas são provadas decidíveis de forma similar. \square

Os sistemas decrementais também satisfazem o Lemma dos pares críticos como será visto, mas para isso será necessário adatar a noção de par crítico para CTRS.

Definição 9.4.4 (Pares Críticos Condicionais) Se $s_1 = t_1 \wedge \dots \wedge s_n = t_n : l \rightarrow r$ e $s'_1 = t'_1 \wedge \dots \wedge s'_{n'} = t'_{n'} : l' \rightarrow r'$ são regras de um CTRS R e l unifica via um unificador mais geral μ com um subtermo não variável de l' , então a equação condicional a seguir é um **par crítico** de R :

$$\langle s_1\mu = t_1\mu \wedge \dots \wedge s_n\mu = t_n\mu \wedge s'_1\mu = t'_1\mu \wedge \dots \wedge s'_{n'}\mu = t'_{n'}\mu : l'\mu[r\mu] = r'\mu \rangle$$

Teorema 9.4.5 (Critério dos Pares Críticos para CTRSs padrão decrementais) Para qualquer CTRS padrão decremental, se cada par crítico é juntável, então o sistema é confluyente e consequentemente convergente.

Demonstração. A prova apresentada em [JW86] para sistemas redutivos pode ser consultada. Adapta-se diretamente para sistemas decrementais. \square

Usando esse critério justifica-se um procedimento de completção de CTRS padrão baseado na inclusão de novas regras para os pares críticos não juntáveis, como apresentado para os sistemas de reescrita não condicionais.

9.5 Contra Exemplos

Apresentam-se contra-exemplos para situações padrão em sistemas não condicionais e na seção seguinte, o procedimento de completção para sistemas condicionais.

Sistemas de reescrita não condicionais são localmente confluentes se todos os seus pares críticos são juntáveis (ainda que não sejam terminantes). Por outro lado, sistemas condicionais não terminantes não precisam ser localmente confluentes, ainda que não existam sobreposições entre as regras. Considere o exemplo a seguir.

$$\begin{array}{l} : b \quad \rightarrow f(b) \\ x = f(x) : f(x) \rightarrow a \end{array}$$

Nesse sistema não existem pares críticos, mas o termo $f(b)$ tem diversas formas normais incluindo $a, f(a), f(f(a)), \dots$. CTRS terminantes sem pares críticos são, outrossim, localmente confluentes [Siv89]. Infelizmente:

Teorema 9.5.1 *Existe um CTRS terminante (mas não decremental) cujos pares críticos são todos juntáveis, mas que não é local confluyente.*

Demonstração. Considere o CTRS R dado pelas regras a seguir.

$$\begin{array}{l} 1. \quad : a \quad \rightarrow b \\ 2. \quad : c \quad \rightarrow k(f(a)) \\ 3. \quad : c \quad \rightarrow k(g(b)) \\ 4. \quad : h(x) \rightarrow k(x) \\ 5. \quad : h(f(a)) \rightarrow c \\ 6. \quad h(f(x)) = k(g(b)) : f(x) \rightarrow g(x) \end{array}$$

Primeiro note que R é terminante (pode ser usada a precedência $h > c > k > f > g > a > b$, e então notar que $l > r$ para cada lado esquerdo l e direito r das regras em R). Todos os pares críticos de R , a saber:

1. $k(f(a)) = k(g(b))$ [2, 3]
2. $k(f(b)) = c$ [1, 5]
3. $k(f(a)) = c$ [4, 5]
4. $h(f(a)) = k(g(b)) : h(g(a)) = c$ [5, 6]

são juntáveis. Mas, o termo $f(a)$ tem duas formas normais: $f(b)$ e $g(b)$. □

9.6 Procedimento de Completação para Sistemas Condicionais

Para CTRSs, a idéia básica de um método de completção para gerar sistemas de reescrita convergentes é a mesma dos sistemas não condicionais. Mas aparecem alguns problemas adicionais; por exemplo, juntabilidade dos pares críticos assegura somente a confluência de sistemas decrementais. Assim,

sempre que se encontra uma equação que não é decremental (o que pode acontecer ainda quando as regras do sistema original o sejam), uma abordagem de completção direta falha. Métodos para manipular equações não decrementais, numa ferramenta de completção, têm sido formulados.

Em [JW86] foi proposta uma técnica utilizando "estreitamento condicional" (a ser estudado no próximo capítulo) junto com completção para detectar alguns casos nos quais pares críticos não decrementais são insatisfáveis (ou mais precisamente, pares críticos cujas condições o são).

Ganzinger [Gan88, Gan91] propôs uma melhora para manusear pares críticos não decrementais e satisfáveis utilizando estreitamento para enumerar as soluções para as quais os pares críticos se satisfazem. Isso permite substituir, em alguns casos, uma equação não decremental por um conjunto de equações decrementais, sem perder nenhuma das consequências da equação original. Sivakumar [Siv89] propôs uma abordagem denominada "esquema de translação" para converter equações condicionais não decrementais em decrementais no contexto da completção utilizando o operador "*if*". Operacionalmente, o efeito é similar à idéia de Ganzinger de enumerar as soluções dos pares críticos, já que isso introduz a habilidade de sobrepor em termos originalmente na condição. Apresentar-se-ão os métodos de completção de Sivakumar em [Siv89].

Para adaptar as técnicas de completção não condicional ao caso condicional modificam-se as regras de inferência da completção não condicional apresentadas na Tabela 8.3.

A regra de "eliminação" (C4) pode ser estendida para eliminar também equações condicionais da forma

$$\dots \wedge s = t \wedge \dots : s = t$$

onde a equação na conclusão também aparece na condição. A regra de "orientação" (C1) deve assegurar não unicamente que o lado esquerdo da regra seja "maior" que o lado direito, mas também que a equação seja decremental. Caso contrário, a juntabilidade dos pares críticos não irá garantir a confluência. A regra de "simplificação" (C3) pode ser estendida para permitir reescrever também termos na condição. A regra de "dedução" (C2) deve ser adaptada para a definição de pares críticos condicionais. Desta forma, obtemos o sistema de regras de inferência apresentado na Tabela 9.2.

Enquanto as equações e regras (condicionais) forem decrementais, a completção procede similar ao caso não condicional. Mas esse não é o caso usual pois, ainda que se inicie com um sistema de equações decrementais, podem ser gerados pares críticos que não o são, o que conduz o procedimento de completção a uma situação de falha.

(C_1)	$\frac{(E \cup \{c : s \dot{=} t\}, R)}{(E, R \cup \{c : s \rightarrow t\})}$	se $s > t, c$	orientação
(C_2)	$\frac{(E, R)}{(E \cup \{c : s = t\}, R)}$	se $\langle c : s = t \rangle \in CP(R)$	dedução
(C_3)	$\frac{(E \cup \{c : s \dot{=} t\}, R)}{(E \cup \{c : u = t\}, R)}$	se $s \rightarrow_R u$	simplificação
	$\frac{(E \cup \{c \wedge p = q : s = t\}, R)}{(E \cup \{c \wedge u = q : s = t\}, R)}$	se $p \rightarrow_R u$	
(C_4)	$\frac{(E \cup \{c : s = s\}, R)}{(E, R)}$		eliminação
	$\frac{(E \cup \{c \wedge s = t : s \dot{=} t\}, R)}{(E, R)}$		

Tabela 9.2: Regras de inferência do procedimento de completção condicional

Exemplo 9.6.1 Considere o sistema de equações condicionais E_0 dado por

1. $even(0) = TT$
2. $odd(s(0)) = TT$
3. $even(s(s(x))) = even(x)$
4. $odd(s(s(x))) = odd(x)$
5. $odd(x) = TT : f(x) = 0$
6. $even(x) = TT : f(x) = s(0)$

As equações definem $f(x)$ como 0 quando x é ímpar e como $s(0)$ quando x é par. Com uma ordem de precedência sobre os operadores selecionada adequadamente (p.ex., $f > odd, even, 0, s; odd, even > TT$), pode-se demonstrar que todas essas equações são decrementais, e estas podem ser orientadas como regras da esquerda para a direita.

Não obstante, as regras 5 e 6 permitem deduzir o seguinte par crítico não decremental:

$$\langle odd(x) = TT \wedge even(x) = TT : s(0) = 0 \rangle$$

Esse é o único par crítico do sistema sendo também insatisfável, já que nenhum valor de x pode ser par e ímpar simultaneamente. Mas a completção falha nesse estágio. A abordagem utilizada em [JW86] para tratar esse problema consiste em buscar soluções para as condições utilizando estreitamento condicional. Dessa forma detectam-se os casos de insatisfabilidade de instâncias das condições dos pares críticos, com relação ao sistema de regras atual. \diamond

Observa-se a completção atuando sobre uma extensão conservativa da teoria original. As equações que são não decrementais podem ser convertidas em equações decrementais na extensão. Se

a completção tem êxito sobre uma nova assinatura, então o sistema de reescrita assim gerado deve também ser convergente para a teoria original.

Seja E um conjunto de equações condicionais usando termos de $Ter(SIG, Var)$. Sejam if, eq e $true$ símbolos de função novos que não estão em SIG e $SIG' = SIG \cup \{if, eq, true\}$. O conjunto E será convertido em E' sobre $Ter(SIG', Var)$ como indicado a seguir. Adicionam-se as equações $eq(x, x) = true$ e $if(true, x) = x$ a E' . Uma equação condicional $c \wedge u = v : s = t$ em E é representada em E' por $c : if(eq(u, v), s) = if(eq(u, v), t)$. Essa **translação** é também reversível, já que pode-se substituir (sempre que necessário) uma equação condicional da forma $c : if(eq(u, v), s) = if(eq(u, v), t)$ por $c \wedge u = v : s = t$.

Essas regras são apresentadas como regras de inferência para o procedimento de completção na Tabela 9.3. Supõe-se que $eq(x, x) = true$ e $if(true, x) = x$ estão incluídas em E .

$(C_5) \quad \frac{(E \cup \{c \wedge u = v : s = t\}, R)}{(E \{c : if(eq(u, v), s) = if(eq(u, v), t)\}, R)} \quad \text{translação}$ $\frac{(E \{c : if(eq(u, v), s) = if(eq(u, v), t)\}, R)}{(E \cup \{c \wedge u = v : s = t\}, R)}$

Tabela 9.3: Regras de inferência de translação para o procedimento de completção condicional

A correção dessas regras de translação segue do fato de que essa é uma extensão conservativa [Siv89].

Teorema 9.6.2 (Correção das regras de translação) *Sejam u e v termos em $Ter(SIG, Var)$ e suponha que $(E_1, R) \vdash (E_2, R)$ usando uma regra de translação. Então $u \leftrightarrow^* v$ é provável em (E_1, R) se, e somente se o é em (E_2, R) .*

Exemplo 9.6.3 No exemplo 9.6.1, o par crítico não decremental

$$\langle odd(x) = TT \wedge even(x) = TT : s(0) = 0 \rangle$$

pode ser substituído por uma equação equivalente decremental na assinatura estendida e, subsequentemente, orientado numa regra decremental (supõe-se que $even$ é "maior" que odd na ordem utilizada):

$$7. \quad odd(x) = TT : if(eq(even(x), TT), s(0)) \rightarrow if(eq(even(x), TT), 0)$$

Continuando com o processo de completção, encontra-se que essa nova regra gerará dois pares críticos. Um deles é:

$$\langle odd(x) = TT : if(eq(even(x), TT), s(0)) = if(eq(even(s(s(x))), TT), 0) \rangle$$

gerado com a regra $even(s(s(x))) \rightarrow even(x)$. Esse par crítico simplifica para uma instância da regra 7 e pode ser eliminado. O outro par crítico, gerado com $even(0) \rightarrow TT$, é:

$$\langle odd(x) = TT : if(eq(TT, TT), s(0)) = if(eq(even(0), TT), 0) \rangle$$

que depois de simplificação, translação e orientação gera a regra decremental:

$$8. : if(eq(odd(0), TT), s(0)) \rightarrow if(eq(odd(0), TT), 0)$$

A completção pára com êxito quando não se tem mais pares críticos e se obtém um sistema de reescrita condicional convergente R_{end} :

i.	$: eq(x, x)$	$\rightarrow true$
ii.	$: if(true, x)$	$\rightarrow x$
1.	$: even(0)$	$\rightarrow TT$
2.	$: odd(s(0))$	$\rightarrow TT$
3.	$: even(s(s(x)))$	$\rightarrow even(x)$
4.	$: odd(s(s(x)))$	$\rightarrow odd(x)$
5.	$odd(x) = TT : f(x)$	$\rightarrow 0$
6.	$even(x) = TT : f(x)$	$\rightarrow s(0)$
7.	$odd(x) = TT : if(eq(even(x), TT), s(0))$	$\rightarrow if(eq(even(x), TT), 0)$
8.	$: if(eq(odd(0), TT), s(0))$	$\rightarrow if(eq(odd(0), TT), 0)$

Esse sistema é convergente e é uma extensão conservativa da teoria equacional inicial E_0 . Assim, para todos os termos u, v na assinatura original, tais que $u \leftrightarrow^* v$ em E_0 , existe uma prova de reescrita $u \rightarrow^* s^* \leftarrow v$ em R_{end} . Para tratar termos da assinatura original, as regras $i, ii, 7, 8$ não são usadas. Dessa forma, essas regras podem ser omitidas e as regras 1-6 formam um sistema de reescrita convergente para decidir o problema de validade em E_0 . \diamond

Outra abordagem para manipular equações não-decrescentes é **reescrita contextual**, sugerida em [Siv89] e [Gan91]. Reescrita contextual é usada para simplificar pares críticos que aparecem durante a completção condicional. Isso não faz unicamente o procedimento mais eficiente, mas também ajuda a concluir com sucesso em muitos casos.

Exemplo 9.6.4 Considere as seguintes regras:

1. $member(x, z) = FF : delete(x, z) \rightarrow z$
2. $different(x, y) = TT : delete(x, y.z) \rightarrow y.delete(x, z)$
3. $different(x, y) = TT : member(x, y.z) \rightarrow member(x, z)$

O par crítico:

$$different(x, y) = TT \wedge member(x, y.z) = FF : y.delete(x, z) = y.z$$

entre as regras 1 e 2 não simplifica diretamente, sempre que não há nenhum sub-termo (na condição da conclusão) que possa ser diretamente reescrito, usando as regras 1, 2 e 3.

Mas não é necessário reescrever o subtermo $member(x, y, z)$ na condição isoladamente. Quando o reescrevemos, podemos supor que a outra condição $different(x, y) = TT$ vale. Se fazemos isso, então a regra 3 aplica e a condição pode ser reescrita no novo contexto, que tem $member(x, z) = FF$, usando a regra para z . O par crítico depois dessas duas simplificações é

$$different(x, y) = TT \wedge member(x, z) = FF : y.z = y.z$$

que é trivial. Assim, usando reescrita contextual, tem-se mostrado que esse par crítico é juntável no sistema. ◇

Denota-se reescrita contextual como \rightarrow_C onde C denota o **contexto**. Um contexto é uma conjugação de equações da forma $p_1 = q_1 \wedge \dots \wedge p_n = q_n$, que são supostas “verdadeiras” quando se faz uma redução. I.e., sempre que tenhamos que verificar se uma condição vale para aplicar uma regra, pode-se supor o contexto verdadeiro para verificar a condição. Segue a definição formal.

Definição 9.6.5 *Seja R um sistema de reescrita condicional. Diz-se que um termo s reescreve no contexto C para um termo s_1 (denotado $s \rightarrow_C s_1$), se uma das seguintes vale:*

1. $s \rightarrow s_1$, ou
2. $s = s_1$ é uma instância de uma equação $p_i = q_i$ em C , e $s \succ s_1$, ou
3. $u_1 = v_1 \wedge \dots \wedge u_n = v_n : l \rightarrow r$ é uma regra condicional em R , σ é uma substituição tal que $l\sigma \equiv s|_{\pi}, u_i \downarrow v_i, \forall i$ sob o mesmo contexto C e $s_1 \equiv s[r\sigma]_{\pi}$.

Ao simplificar uma equação condicional, reescrita contextual adiciona mais poder. Quando um subtermo é reduzido no conseqüente, o contexto é formado por todas as equações na condição. Quando um subtermo é reescrito na condição, as outras equações na condição formam o contexto. Na Tabela 9.4 apresentam-se as regras de inferência para simplificação.

$(C'_3) \quad \frac{(E \cup \{c : s \doteq t\}, R)}{(E \cup \{c : u = t\}, R)} \quad \text{se } s \rightarrow_c u \quad \text{simplificação}$
$\frac{(E \cup \{c \wedge p = q : s \doteq t\}, R)}{(E \cup \{c \wedge u = q : u = t\}, R)} \quad \text{se } p \rightarrow_c u$

Tabela 9.4: Regras de inferência de simplificação para o procedimento de completção condicional

A correção dessa regra de simplificação é expressa na seguinte proposição.

Teorema 9.6.6 *Seja $\langle E, R \rangle \vdash \langle E', R \rangle$ uma derivação utilizando simplificação. Então, $u \leftrightarrow_{E \cup R}^* v$ se, e somente se $u \leftrightarrow_{E' \cup R}^* v$.*

Para regras da forma $c : if(c_1, l) \rightarrow if(c_1, r)$, obtidas por translação, é suficiente considerar sobreposições em c_1 unicamente. A idéia principal para demonstrar que essa restrição sobre pares críticos é completa, é que se a regra $c : if(c_1, l) \rightarrow if(c_1, r)$ é usada numa prova de igualdade entre dois termos s e t em $Term(\Sigma, Var)$, então ela deve ser usada com uma substituição σ para a qual $c_1\sigma \rightarrow^* true$. De outra forma, o operador if irá persistir na prova. Assim, deverá ser o caso que $if(c_1, l)\sigma \rightarrow^* if(true, l\sigma) \rightarrow l\sigma$. Em qualquer prova dessa forma, conseqüentemente, existirá um pico da forma?

$$l\sigma^* \leftarrow if(c_1\sigma, l\sigma) \rightarrow if(c_1\sigma, r\sigma)$$

onde a reescrita em $c_1\sigma$ é um pico crítico. Se todos os picos críticos em c_1 são juntáveis em R , então existe uma prova que pode ser feita “menor” usando alguma extensão da ordem decrescente de R para uma ordem sobre provas.

Uma equação $c : s = t$ pode ser eliminada se $c' : s' = t' \in E$ e $c'\sigma : s'\sigma = t'\sigma \equiv c : s = t$, para uma substituição σ . Isso é denominado *subsumption* e a correspondente regra de inferência apresentada na Tabela 9.5.

$(C_6) \quad \frac{(E \cup \{e_1 \equiv c : s \doteq t, e_2 \equiv c' : s' \doteq t'\}, R)}{(E \cup \{c : s = t\}, R)} \quad \text{se } \exists \sigma, e_1\sigma \equiv e_2 \quad \text{subsumption}$
--

Tabela 9.5: Regra de inferência de *subsumption* para o procedimento de completção condicional

Uma abordagem diferente para tratar equações não-decrescentes é proposta por Ganzinger. O seu método consiste em enumerar instâncias de superposição. Usa-se estreitamento condicional sobre condições de equações, que, em caso de um sistema R confluyente, é um procedimento completo para computar as soluções de equações condicionais ([JW86]). Na maioria dos casos, não obstante, essa técnica *per se* não irá levar a um procedimento de completção terminante independentemente. Ele deverá ser acompanhado de técnicas suficientemente fortes para eliminação de equações por provas limitadas (por exemplo, por reescrita contextual ou inferências de *subsumption*).

Dadas uma equação condicional $C \wedge u = v : s = t$ e uma regra condicional $D : l \rightarrow r$. Seja π uma ocorrência não-variável em $u = v$ tal que $\sigma = mgu((u = v)|_\pi, l)$. Adicionalmente, se $u\sigma > v\sigma$ então π está dentro de u , e se $v\sigma > u\sigma$, π está dentro de v . Com essas suposições, $D\sigma \wedge C\sigma \wedge (u = v)[r]_\pi\sigma : s\sigma = t\sigma$ é chamada uma **instância de superposição** para superpor

$D : l \rightarrow r$ sobre $u = v$ em $C \wedge u = v : s = t$.

A seguinte proposição estabelece a correção ao incluir instâncias de sobreposição.

Teorema 9.6.7 *Dados E e R conjuntos de equações e regras. Seja $C \wedge u = v : s = t$ uma equação em E . Suponha que E contém todas as instâncias da equação gerada por sobreposição de cada regra em $R \cup x = x \rightarrow \text{true}$ sob condição $u = v$ na equação dada. Se $s\sigma \leftrightarrow_{E,R} t\sigma$ usando a equação $C \wedge u = v : s = t$, então tem-se também uma prova de $s\sigma = t\sigma$ que é mais simples que $s\sigma \leftrightarrow_{E,R} t\sigma$.*

Demonstração. (Para uma prova formal veja [Gan91]). Tem-se que proveer simplificações de provas que aplicam uma equação sobre uma substituição σ tal que as provas para qualquer das correspondentes instâncias das condições $u = v$ são provas de reescrita. Uma prova de reescrita para $u\sigma = v\sigma$ é ou vazia (i.e., $u\sigma \equiv v\sigma$) ou irá conter algum passo de reescrita de $u\sigma$ ou $v\sigma$. No primeiro caso, a meta-regra $x = x \rightarrow \text{true}$ pode ser superposta sobre $u = v$. Isso irá eliminar a condição da equação. Se se adiciona ao conjunto de equações essa instância de sobreposição, usando a nova equação com uma condição menos para provar $s\sigma = t\sigma$ tem-se menor complexidade. No segundo caso, o primeiro passo de reescrita pode estar ou dentro de σ ou o redex sobrepõe com uma posição não variável em, seja, u . No primeiro desses casos, reescreve-se primeiro σ para σ' e então, usa-se a mesma equação com a substituição reduzida σ' para provar $s\sigma' = t\sigma'$, seguida de *unfolding* σ' para σ é menos complexo. Essi é justificado principalmente porque a aplicação de uma equação usando uma substituição menor é menos complexa. O último caso, pode ser simplificado se a instância de sobreposição que corresponde à sobreposição do redex do primeiro passo de reescrita com u é adicionado às equações (o que é suposto). Nesta instância, estreita-se u para z , e $u\sigma \equiv u\tau\tau' \rightarrow z\tau'$. Ao aplicar a equação dada sob substituição σ pode então ser substituída aplicando a instância de sobreposição sob substituição τ' . O que é menos complexo, sempre que $u\sigma > z\tau'$. \square

9.7 Exemplos de completção

Nesta seção ilustra-se a aplicação das regras de inferência para completção de especificações condicionais.

Exemplo 9.7.1 O primeiro exemplo formaliza a ordem $<$ sobre números naturais como tratada por Sivakumar em [Siv89].

1. $0 < s(0) = TT$
2. $s(x) < s(y) = x < y$
3. $x < y = TT \wedge y < z = TT : x < z = TT$

As duas primeiras equações são orientáveis de esquerda para direita como regras de reescrita decrescentes. A terceira equação (transitividade) não é decrescente. Ela tem uma variável

“extra” y na condição que não está em nenhum lado da conclusão. Aplicação de translação produz a seguinte regra:

$$3. y < z = TT : if(eq(x < y, TT), x < z) \rightarrow if(eq(x < y, TT), TT)$$

Para esta regra ir-se-á sobrepor unicamente no subtermo $eq(x < y, TT)$ do lado esquerdo segundo os linhamentos da seção precedente. O par crítico não decrescente com regra 1 é:

$$s(0) < z = TT : if(eq(TT, TT), 0 < z) = if(eq(0 < s(0), TT), TT) [1, 3]$$

Depois de simplificação e translação, obtem-se a regra:

$$s(y) < z = TT : if(eq(x < y, TT), s(x) < z) = if(eq(s(x) < s(y), TT), TT) [2, 3]$$

Logo de simplificação e duas translações (primeiro para mover o termo $eq(x < y, TT)$ para a condição, e segundo para mover $s(y) < z$ para a conclusão), obtem-se a regra:

$$5. x < y = TT : if(eq(s(y) < z, TT), s(x) < z) \rightarrow if(eq(s(y) < z, TT), TT)$$

A regra 4 tem unicamente um par crítico com a regra 2:

$$: if(eq(0 < z, TT), 0 < s(z)) = if(eq(s(0) < s(z), TT), TT) [4, 2]$$

Subsequente simplificação e translação levam à seguinte regra condicional:

$$6. 0 < z = TT : 0 < s(z) \rightarrow TT$$

Essa translação para uma regra condicional é essencial para a terminação do procedimento de completção. Ela elimina futuras sobreposições no termo $0 < z$ term, que não ocorre na condição.

Regras 5 e 2 produzem o seguinte par crítico:

$$x < y = TT : if(eq(y < z, TT), s(x) < s(z)) = if(eq(s(y) < s(z), TT), TT) [5, 2]$$

Por simplificação e translação para

$$y < z = TT : if(eq(x < y, TT), x < z) = if(eq(x < y, TT), TT)$$

a condição $if(eq(x < y, TT), x < z)$ pode ser reescrita no contexto $y < z = TT$ pela regra 3. Isso leva à equação trivial, e conseqüentemente esse par crítico é juntável.

Regras 6 e 1 produzem o seguinte par crítico:

$$0 < 0 = TT : TT = TT [6, 1]$$

Esse par crítico é trivial, e com regras 6 e 3 obtem-se?

$$0 < y = TT \wedge s(y) < z = TT : if(eq(TT, TT), 0 < z) = if(eq(0 < s(y), TT), TT) [6, 3]$$

Esse par crítico é também juntável. Primeiro, reescreve-se $0 < s(y)$ para TT no contexto $0 < y = TT \wedge s(y) < z = TT$ usando a regra 6. Então, simplificação por $eq(x, x) \rightarrow true$ e $if(true, x) \rightarrow x$ produz:

$$0 < y = TT \wedge s(y) < z = TT : 0 < z = TT$$

que pode ser transladada para:

$$0 < y = TT : if(eq(s(y) < z, TT), 0 < z) = if(eq(s(y) < z, TT), TT)$$

O último é uma instância da regra 5 e, conseqüentemente, pode ser simplificada a uma equação trivial.

O procedimento de completção pará com as seguintes regras:

- | | | |
|-----|----------------|---|
| i. | : | $eq(x, x) \rightarrow true$ |
| ii. | : | $if(true, x) \rightarrow x$ |
| 1. | : | $0 < s(0) \rightarrow TT$ |
| 2. | : | $s(x) < s(y) \rightarrow x < y$ |
| 3. | $y < z = TT :$ | $if(eq(x < y, TT), x < z) \rightarrow if(eq(x < y, TT), TT)$ |
| 4. | : | $if(eq(s(0) < z, TT), 0 < z) \rightarrow if(eq(s(0) < z, TT), TT)$ |
| 5. | $x < y = TT :$ | $if(eq(s(y) < z, TT), s(x) < y) \rightarrow if(eq(s(y) < z, TT), TT)$ |
| 6. | $0 < z = TT :$ | $0 < s(z) \rightarrow TT$ |

Eliminação das regras “if” (e das regras i e ii) leva ao sistema 1, 2 e 6, que é convergente e equivalente ao sistema equacional original sobre a assinatura inicial. \diamond

Exemplo 9.7.2 Este exemplo compara as abordagens transformacional e via instâncias de sobreposição. O exemplo é uma formalização das relação de ordem nos inteiros com zero, sucessor e precessor $0, s, p, <$ tomado de [Kap84b] e apresentado por Ganzinger em [Gan91].

- | | | |
|----|----------------|-----------------------|
| 1. | : | $0 < 0 = FF$ |
| 2. | : | $0 < s(0) = TT$ |
| 3. | : | $s(x) < y = x < p(y)$ |
| 4. | : | $p(x) < y = x < s(y)$ |
| 5. | $y < x = TT :$ | $y < s(x) = TT$ |
| 6. | $y < x = FF :$ | $y < p(x) = FF$ |
| 7. | : | $s(p(x)) = x$ |
| 8. | : | $p(s(x)) = x$ |

As equações são orientáveis como regras decrescentes de esquerda para direita. Obtem-se dois pares críticos não decrementais ao sobrepor as regras 4 e 6 e 3 e 5, respectivamente?

- | | | |
|-----|-------------------|---------------------|
| 9. | $y < s(x) = FF :$ | $y < x = FF$ [4, 6] |
| 10. | $y < p(x) = TT :$ | $y < x = TT$ [3, 5] |

Para a equação 9, por exemplo, a sobreposição das regras (1-8) sobre suas condições leva às seguintes equações:

$$11. \quad y < x = FF : y < p(x) = FF$$

de sobrepor $s(p(x)) \rightarrow x$.

$$12. \quad y < x = TT \wedge TT = FF : y < x = FF$$

de sobrepor $y < x = TT : y < s(x) \rightarrow TT$.

$$13. \quad TT = FF : 0 < 0 = FF$$

de sobrepor $0 < s(0) \rightarrow TT$.

$$14. \quad x < p(s(y)) = FF : x < p(y) = FF$$

de sobrepor $s(x) < y \rightarrow x < p(y)$.

$$15. \quad x < s(s(y)) = FF : x < s(y) = FF$$

de sobrepor $p(x) < y \rightarrow x < s(y)$.

Então devem-se considerar as sobreposições de instâncias geradas. A equação 11 é reduzida (contextualmente) pela regra 6. A equação 12 tem uma condição insatisfazível. Isso pode ser detectado considerando-a como uma equação não decrescente e observando que não existem sobreposições sobre $TT = FF$. Similarmente, a equação 13 é trivial. Para eliminar a equação 14 primeiro reduz-se a condição para $x < y = FF$ pela regra 8 e então a condição é reduzida contextualmente pela regra 6 para $FF = FF$. Finalmente, a equação 15 é subsumida pela equação 9 (equação da qual a equação 15 foi obtida).

De forma similar pode ser provada a convergência da equação não decrescente 10 (exercício 9.5).

O procedimento converge e finaliza com R como o sistema de equações original, orientadas de esquerda para direita e com E consistente das duas equações não decrescentes 9 e 10. As últimas, foram provadas irrelevantes para a teoria equacional.

Utilizando a abordagem transformacional, as equações não decrescentes 9 e 10 são transformadas respectivamente nas regras:

$$\begin{aligned} 9'. & \quad : if(eq(y < s(x), FF), y < x) \rightarrow if(eq(y < s(x), FF), FF) \\ 10'. & \quad : if(eq(y < p(x), TT), y < x) \rightarrow if(eq(y < p(x), TT), TT) \end{aligned}$$

Então, obtém-se novos pares críticos com, por exemplo, a regra $9'$, que correspondem às instâncias de sobreposição 11-15, sempre que sobreposição é unicamente permitida sobre o termo $eq(y < s(x), FF)$.

Para a regra $9'$, por exemplo, obtém-se os seguintes pares críticos por sobreposição das regras (1-8):

$$11'. \quad : if(eq(y < x, FF)y < p(x)) = if(eq(y < s(p(x)), FF), FF)$$

de sobrepor $s(p(x)) \rightarrow x$. Por simplificação com $s(p(x)) \rightarrow x$ e translação, obtém-se a equação $y < x = FF : y < p(x) = FF$ (11!), que pode ser simplificada contextualmente como $y < x = FF : FF = FF$ com a regra 6 e logo eliminada como uma equação trivial.

$$12'. \quad y < x = TT : if(eq(TT, FF), y < x) = if(eq(y < s(x), FF), FF)$$

de sobrepor $y < x = TT : y < s(x) \rightarrow TT$. Esta pode ser simplificada contextualmente como $y < s(x) \rightarrow_{y < x = TT} TT$ com a mesma regras e então transladada, obtendo a equação $y < x = TT \wedge TT = FF : y < x = FF$ que é insatisfazível.

$$13'. \quad : if(eq(TT, FF), 0 < 0) = if(eq(0 < s(0), FF)FF)$$

de sobrepor $0 < s(0) \rightarrow TT$. A equação 13' pode ser simplificada com a mesma regra e então transladada para obter a equação insatisfazível $TT = FF : 0 < 0 = FF$ (13!) que pode ser eliminada.

$$14'. \quad : if(eq(y < p(s(x)), FF), y < p(x)) = if(eq(s(y) < s(x), FF)FF)$$

de sobrepor $s(x) < y \rightarrow x < p(y)$. Dois passos de simplificação e uma translação levam à equação $Simy < x = FF : y < p(x) = FF$ (14!) que pode ser orientada, mas coincide com a regra 6.

$$15'. \quad : if(eq(x < s(s(y)), FF), x < s(y)) = if(eq(p(x) < s(y), FF)FF)$$

de sobrepor $p(x) < y \rightarrow x < s(y)$. Duas simplificações e uma orientação levam a regra $if(eq(y < s(s(x)), FF), y < s(x)) \rightarrow if(eq(y < s(s(x)), FF), FF)$ que é uma instância da regra $9'$ e assim pode ser eliminada.

Similarmente, pelo método translacional, pares críticos com a regra $10'$ são eliminados.

Dessa forma o processo de completção com esta abordagem finaliza com o sistema de regras que consiste das regras i,ii, 1-8, $9'$ e $10'$. \diamond

Exercícios do Capítulo 9

Exercício 9.1 Consulte e complete os detalhes da prova do Teorema 9.4.1.

Exercício 9.2 Complete os detalhes da prova do Teorema de decidibilidade das propriedades básicas dos sistemas condicionais decrementais 9.4.3.

Exercício 9.3 Consulte e complete os detalhes da prova do critério dos pares críticos para sistemas condicionais padrão decrementais 9.4.5.

Exercício 9.4 Considere o seguinte sistema de reescrita de termos (programa) para ordenar listas utilizando o bem conhecido método de fusão de listas, *merge sort*.

- [1] $split(nil, nil, a.nil) \rightarrow a.nil$
- [2] $split(l_1, l_2, a.b.l) \rightarrow split(a.l_1, b.l_2, l)$
- [3] $split(l_1, l_2, a.nil) \rightarrow split(a.l_1, l_2, nil)$
- [4] $split(l_1, l_2, nil) \rightarrow merge(split(nil, nil, l_1), split(nil, nil, l_2))$
- [5] $merge(nil, l_1) \rightarrow l_1$
- [6] $merge(l_1, nil) \rightarrow l_1$
- [7] $a \leq b \quad : \quad merge(a.l_1, b.l_2) \rightarrow a.merge(l_1, b.l_2)$
- [8] $a > b \quad : \quad merge(a.l_1, b.l_2) \rightarrow b.merge(a.l_1, l_2)$
- [9] $mergesort(l) \rightarrow split(nil, nil, l)$

O sistema gerá os seguintes quatro pares críticos:

- [8, 7] $\langle (a \leq b \wedge a > b) \quad : \quad b.merge(a.l_1, l_2), a.merge(l_1, b.l_2) \rangle$
- [7, 8] $\langle (a > b \wedge a \leq b) \quad : \quad a.merge(l_1, b.l_2), b.merge(a.l_1, l_2) \rangle$
- [3, 1] $\langle split(a.nil, nil, nil), a.nil \rangle$
- [1, 3] $\langle a.nil, split(a.nil, nil, nil) \rangle$

Sendo as condições dos pares críticos das regras 7 e 8 insatisfazíveis, estes podem ser desconsiderados. Por outra parte dos últimos pares críticos pode-se criar uma regra adicional:

$$[10] \quad split(a.nil, nil, nil) \rightarrow a.nil \quad [1, 3]$$

1. A inclusão desta nova regra soluciona os problemas? Obtém-se um sistema de regras convergente?
2. Continue o processo de completção.

Exercício 9.5 Complete todos os detalhes do processo de completção do exemplo 9.7.1.

Exercício 9.6 Complete o processo de completção do exemplo 9.7.2

1. via a abordagem de instâncias de sobreposição do exemplo 9.7.2. Verifique, a convergência da equação não decrescente 10.
2. Complete o processo utilizando a abordagem transformacional. Verifique que todos os pares críticos gerados por sobreposição com a regra $10'$ podem ser eliminados.