

Modelos Abstractos de Cómputo

Curso 2009-20010

Plan de sesiones a cargo de Mario Rodríguez Artalejo

1. 30 Nov: Reescritura condicional

1.1. Sistemas de reescritura condicionales

Referencias básicas: [29], sección 2.6 y [5], sección 11.3.

Tarea: Ejercicios 1 y 2.

1.2. Condiciones de c -convergencia

Referencia básica: [29], sección 4.1.

1.3. Funciones indeterministas, *call time choice*

Referencia básica: [29], sección 4.2.

Tarea: Ejercicio 3.

1.4. Reescritura condicional con semántica CRWL

Referencia complementaria: [29], secciones 4.3, 4.4 y 4.5.

Tarea: Ejercicios 4 y 5.

2. 14 Dic: Estrechamiento

2.1. Unificación módulo una teoría ecuacional

Referencias básicas: [29], sección 2.2, y [5], sección 10.1.

Tarea: Ejercicio 6.

2.2. El estrechamiento como procedimiento de \mathcal{R} -unificación

Referencia básica: [29], sección 5.1.

Referencia complementaria: [5], sección 11.6.

Tarea: Ejercicio 7.

2.3. Corrección y completitud del estrechamiento

Referencia básica: [29], sección 5.2.

Referencia complementaria: [19].

Tarea: Ejercicio 8.

2.4. Estrategias de estrechamiento

Referencias complementarias: [29], sección 5.3 y [23].

3. 21 Dic: Estrechamiento condicional

3.1. El estrechamiento condicional y su corrección

Referencia básica: [29], sección 5.4.

Tarea: Ejercicio 9.

3.2. Resultados limitados de completitud

Referencia básica: [29], sección 5.5.

Tarea: Ejercicio 10.

3.3. Unificación módulo un SRC con semántica CRWL

Referencia básica: [29], sección 5.5.

Tarea: Ejercicio 11.

4. 11 Ene: Estrechamiento perezoso

4.1. El cálculo ingenuo SNC

Referencia básica: [29], sección 6.1.

Tarea: Ejercicio 12.

4.2. El cálculo LNC_d orientado al determinismo

Referencia básica: [29], sección 6.2.

Referencias complementarias: [24, 25].

Tarea: Ejercicio 13.

4.3. El cálculo CLNC orientado a la semántica CRWL

Referencia básica: [29], sección 6.3.

Referencias complementarias: [12, 27].

Tarea: Ejercicio 14.

4.4. Comparación entre LNC_d y CLNC

Referencia básica: [29], sección 6.4.

5. 18 Ene: Estrechamiento necesario

5.1. La estrategia NN para ISSs

Referencia básica: [29], sección 7.1.

Referencia complementaria: [3].

Tarea: Ejercicio 15.

5.2. Corrección y completitud de NN

Referencia básica: [29], sección 7.2.

Referencia complementaria: [3].

Tarea: Ejercicio 16.

5.3. Optimalidad de NN y comparación con otras estrategias

Referencia básica: [29], secciones 7.3 y 7.4.

Referencia complementaria: [3].

Tarea: Ejercicio 17.

5.4. Extensión de NN a SRCs con semántica CRWL

Referencias complementarias: [29], capítulo 8 y [30].

Ejercicios

1. Dadas dos listas xs e ys , su *diferencia* $xs \setminus ys$ está definida siempre que ys sea un prefijo de xs , en cuyo caso $xs \setminus ys$ es el sufijo de xs que queda al eliminar el prefijo ys . Se pide construir un SRC basado en constructoras que incluya reglas de reescritura para la operación \setminus , y mostrar el cálculo de la forma normal del término correspondiente a la diferencia de listas $[0, 1, 2] \setminus [0, 1]$ (suponiendo que 0, 1 y 2 sean tres constantes).
Pista: Además de utilizar las constructoras de listas $[]$ y $;$, conviene que el SRC pedido incluya reglas de reescritura para la operación de concatenación de listas $++$.
2. Suponiendo que los números naturales se representen como términos construidos con las dos constructoras 0 y s , se pide construir un SRC basado en constructoras que incluya (entre otras) reglas para una operación fib , tal que para todo $n \in \mathbb{N}$ la forma normal del término $fib(s^n(0))$ sea calculable en $\mathcal{O}(n)$ pasos de reescritura y represente el número de Fibonacci de lugar n .
Pista: Conviene que el SRC pedido incluya reglas de reescritura para una operación auxiliar $dosFib$, tal que para todo $n \in \mathbb{N}$, la forma normal del término $dosFib(s^n(0))$ represente la pareja formada por el número de Fibonacci de lugar n y el número de Fibonacci de lugar $n + 1$.

3. Sea \mathcal{R} el SRT basado en constructoras formado por las siguientes reglas de reescritura:

$$\text{head}(x : xs) \rightarrow x$$

$$\text{take}(0, xs) \rightarrow []$$

$$\text{take}(s(n), []) \rightarrow []$$

$$\text{take}(s(n), x : xs) \rightarrow x : \text{take}(n, xs)$$

$$\text{from}(x) \rightarrow x : \text{from}(s(x))$$

$$x + 0 \rightarrow x$$

$$x + s(y) \rightarrow s(x + y)$$

$$\text{double}(x) \rightarrow x + x$$

$$\text{coin} \rightarrow 0$$

$$\text{coin} \rightarrow s(0)$$

Considera el término $e \equiv \text{head}(\text{from}(\text{double}(\text{coin})))$. Explica intuitivamente:

- a) $e \rightarrow_{\mathcal{R}}^* 0$ es posible usando *call time choice*.
- b) $e \rightarrow_{\mathcal{R}}^* s(0)$ no es posible usando *call time choice*.
- c) $e \rightarrow_{\mathcal{R}}^* s(s(0))$ es posible usando *call time choice*.

4. Sean \mathcal{R} y e como en el ejercicio anterior. Comprueba utilizando la definición formal de $\rightarrow_{\mathcal{R}, \text{CRWL}}$:

- a) $e \rightarrow_{\mathcal{R}, \text{CRWL}}^* 0$ es posible.
- b) $e \rightarrow_{\mathcal{R}, \text{CRWL}}^* s(0)$ no es posible.
- c) $e \rightarrow_{\mathcal{R}, \text{CRWL}}^* s(s(0))$ es posible.

5. Sea \mathcal{R} el SRC basado en constructoras formado por las siguientes reglas de reescritura:

$$\text{azar}(x) \rightarrow x$$

$$\text{azar}(x) \rightarrow s(x)$$

$$\text{colega}(x) \rightarrow y \Leftarrow s(\text{azar}(x)) == \text{azar}(y)$$

$$\text{repite}(x) \rightarrow x : \text{repite}(x)$$

$$azares(x) \rightarrow repite(azar(x))$$

$$trunca(x : y : zs) \rightarrow x : y : []$$

Justifica razonadamente las afirmaciones siguientes:

a) $colega(x) \rightarrow_{\mathcal{R}, CRWL}^* x$ es posible.

b) $azares(x) \rightarrow_{\mathcal{R}, CRWL}^* x : x : \perp$ es posible.

c) $azares(x) \rightarrow_{\mathcal{R}, CRWL}^* s(x) : s(x) : \perp$ es posible.

d) $azares(x) \rightarrow_{\mathcal{R}, CRWL}^* x : s(x) : \perp$ no es posible.

e) $azares(x) \rightarrow_{\mathcal{R}, CRWL}^* s(x) : x : \perp$ no es posible.

f) $trunca(azares(x)) \rightarrow_{\mathcal{R}, CRWL}^* t_0 : t_1 : []$

es posible en los tres casos siguientes y en ningún otro: cuando t_0, t_1 son ambos \perp , cuando t_0, t_1 son ambos x , y cuando t_0, t_1 son ambos $s(x)$.

6. Sea \mathcal{R} un SRT. Se dice que una sustitución σ es un \mathcal{R} -unificador de una ecuación entre términos $s \approx t$ si $\sigma(s)$ y $\sigma(t)$ son equivalentes módulo la teoría ecuacional asociada a \mathcal{R} (en símbolos, $\sigma(s) \approx_{\mathcal{R}} \sigma(t)$). En cada uno de los apartados siguientes se pide construir un SRT \mathcal{R} basado en constructoras, una sustitución σ , y dos términos s y t que no sean c-términos, de forma que σ sea un \mathcal{R} -unificador de $s \approx t$ y además se cumplan las condiciones adicionales indicadas en cada caso.

a) σ es *c-normalizada y cerrada*, en el sentido de que para toda variable x que aparezca en $s \approx t$ se cumple que $\sigma(x)$ es un c-término cerrado.

b) σ es *c-normalizada*, en el sentido de que para toda variable x que aparezca en $s \approx t$ se cumple que $\sigma(x)$ es un c-término. Pero σ no es cerrada (i.e., para alguna de las variables x que aparecen en $s \approx t$, se cumple que $\sigma(x)$ no es cerrado).

c) σ es *normalizada*, en el sentido de que para toda variable x que aparezca en $s \approx t$ se cumple que $\sigma(x)$ es un término en forma normal con respecto a \mathcal{R} . Pero σ no es c-normalizada (i.e., para alguna de las variables x que aparecen en $s \approx t$, se cumple que $\sigma(x)$ no es un c-término).

7. Sea \mathcal{R} el SRT formado por las reglas:

$$s(p(x)) \rightarrow x$$

$$p(s(x)) \rightarrow x$$

$$x + 0 \rightarrow x$$

$$x + s(y) \rightarrow s(x + y)$$

$$x + p(y) \rightarrow p(x + y)$$

Dado el objetivo ecuacional $s(x + y) \approx^? p(x + z)$, se pide:

- a) Comprobar que la sustitución $\sigma = \{y \mapsto p(y_1), z \mapsto s(y_1)\}$ se puede calcular como solución del objetivo, usando estrechamiento.
 - b) Comprobar que σ es una solución correcta, viendo que $\sigma(s(x + y)) \downarrow_{\mathcal{R}} \sigma(p(x + z))$.
8. Sea $\theta = \{x \mapsto 0, y \mapsto p(0 + u), z \mapsto s(0 + u)\}$, donde u es una nueva variable. Nótese que θ es una sustitución normalizada con respecto al SRT \mathcal{R} del ejercicio anterior. Se pide:
- a) Verificar que θ es una solución correcta para el objetivo ecuacional del ejercicio anterior, comprobando que $\theta(s(x + y) \approx^? p(x + z)) \rightarrow_{\mathcal{R}^*}^* \underline{t}$.
 - b) Aplicar la construcción del lema de ascensión de Hullot (ver [29], pg. 83, Lema 5.2.2) para elevar la secuencia de reducción $\theta(s(x + y) \approx^? p(x + z)) \rightarrow_{\mathcal{R}^*}^* \underline{t}$ a una secuencia de estrechamiento $s(x + y) \approx^? p(x + z) \rightsquigarrow_{\mathcal{R}^*, \sigma}^* \underline{t}$ que calcule la solución σ del ejercicio anterior. Obsérvese que $\sigma \preceq \theta$.
 - c) Razonar que $\theta' = \{x \mapsto 0 + 0, y \mapsto p(s(p(0)) + (u + 0)), z \mapsto s(p(s(0)) + (u + 0))\}$ es una sustitución no normalizada pero sí normalizable con respecto a \mathcal{R} . Calcular la forma normalizada $\theta' \downarrow_{\mathcal{R}}$ de θ' , y utilizar el apartado anterior para demostrar que la solución σ calculada por estrechamiento es más general que θ' módulo \mathcal{R} .

9. Sea \mathcal{R} el SRC formado por las reglas

$$\begin{aligned} \text{even}(0) &\rightarrow \text{true} \\ \text{even}(s(x)) &\rightarrow \text{odd}(x) \\ \\ \text{odd}(x) &\rightarrow \text{true} \quad \Leftarrow \quad \text{even}(x) \approx \text{false} \\ \text{odd}(x) &\rightarrow \text{false} \quad \Leftarrow \quad \text{even}(x) \approx \text{true} \end{aligned}$$

y sea S el objetivo ecuacional $\text{even}(s(y)) \approx^? \text{true}$. Se pide:

- a) Estudiar el cómputo mediante estrechamiento condicional que aparece en el Ejemplo 5.4.2 de [29] (pg. 90), que calcula la solución $\sigma = \{y \mapsto s(0)\}$ para S .
 - b) Comprobar que la solución calculada es correcta, verificando que $\sigma(S) \rightarrow_{\mathcal{R}^*}^* \underline{t}$. Observar la correspondencia entre los pasos de reescritura de esta secuencia de reducción y los pasos de estrechamiento ejecutados para el cómputo de σ .
10. Sea \mathcal{R} el SRC de tipo 1 (esto es, sin variables extra) formado por las reglas:
- $$\begin{aligned} \text{evenLength}([]) &\rightarrow \text{true} \\ \text{evenLength}(x : xs) &\rightarrow \text{oddLength}(xs) \end{aligned}$$

$$\begin{aligned} \text{oddLength}(xs) &\rightarrow \text{true} && \Leftarrow && \text{evenLength}(xs) \approx \text{false} \\ \text{oddLength}(xs) &\rightarrow \text{false} && \Leftarrow && \text{evenLength}(xs) \approx \text{true} \end{aligned}$$

Sea además S el objetivo ecuacional $\text{evenLength}(x : ys) \approx^? \text{true}$. Dada la sustitución $\theta = \{x \mapsto s(x'), ys \mapsto 0 : []\}$, se pide:

- a) Verificar que θ es una solución de S módulo \mathcal{R} , comprobando que $\theta(S) \mapsto_{\mathcal{R}}^* \underline{t}$, siendo $\mapsto_{\mathcal{R}}$ la relación de reescritura entre objetivos que añade en cada paso las condiciones de la instancia de regla aplicada al nuevo objetivo (ver [29], Definición 5.5.1, pg. 92).
 - b) Aplicar la construcción del lema de ascensión para SRCs de tipo 1 (Lema 5.5.3 en la pg. 92 de [29]) a la secuencia de reducción del apartado anterior, obteniendo una secuencia de estrechamiento $S \rightsquigarrow_{\mathcal{R}, \sigma}^* \underline{t}$ que calcule una solución σ de S tal que $\sigma \preceq \theta$.
11. Estudiar los contraejemplos contruidos por Giovanetti y Moiso para demostrar la incompletitud del estrechamiento condicional en el caso de SRCs con variables extra (ver Ejemplos 5.5.5 y 5.5.8 en [29], pp. 94 y 95). Demostrar que estos contraejemplos dejan de ser válidos si se utiliza la semántica CRWL en lugar de la semántica ecuacional.
Pista: Suponiendo que \mathcal{R} sea un SRC basado en constructoras, las soluciones de problemas de \mathcal{R} -unificación válidas con respecto a la semántica ecuacional no siempre son válidas con respecto a la semántica CRWL.
12. Sea \mathcal{R} el SRC basado en constructoras presentado en el ejercicio 5. Considera el objetivo $S : \text{trunca}(\text{azares}(x)) =?= x : xs$ y las dos sustituciones $\sigma_1 = \{xs \mapsto x : []\}$ y $\sigma_2 = \{xs \mapsto s(x) : []\}$. Muestra que el cálculo ingenuo de estrechamiento SNC definido en la sección 6.1 de [29] puede obtener cualquiera de estas dos sustituciones como respuesta calculada para S utilizando \mathcal{R} como programa. ¿Es σ_1 una respuesta correcta con respecto a la semántica CRWL? ¿Lo es σ_2 ?
13. Sea LNC_d el cálculo de estrechamiento perezoso explicado en la Sección 6.2 de [29].
- a) Reformula las reglas de transformación de objetivos de este cálculo que plantean pasos de estrechamiento con reglas de reescritura del SRT utilizado - a saber, las tres reglas (IZ), (DR) y $(EM)_P$ - de manera que se puedan aplicar también en el caso de un SRC basado en constructoras y con condiciones de c-convergencia.
 - b) Sean \mathcal{R} , S , σ_1 y σ_2 como en el ejercicio 12. Muestra que LNC_d reformulado del modo planteado en el apartado anterior puede calcular tanto σ_1 como σ_2 como respuesta para S utilizando \mathcal{R} como programa. ¿Es LNC_d correcto con respecto a la semántica CRWL?
14. Sean \mathcal{R} , S , σ_1 y σ_2 como en los dos ejercicios anteriores.

- a) Muestra que el cálculo de estrechamiento perezoso CLNC explicado en la sección 6.3 de [29] solo puede calcular una de las dos sustituciones como respuesta para S utilizando \mathcal{R} como programa.
- b) Según el apartado anterior y los resultados del ejercicio 13, hay una sustitución σ tal que LNC_d puede calcular σ como respuesta para S pero CLNC no puede hacerlo (utilizando \mathcal{R} como programa en ambos casos). Localiza alguna regla de transformación de objetivos de LNC_d que sea responsable de esta diferencia de comportamiento entre los dos cálculos.

15. Sea \mathcal{R} el SRT basado en constructoras formado por las reglas

$$\begin{aligned}
\text{init}(x : []) &\rightarrow [] \\
\text{init}(x : (y : zs)) &\rightarrow x : \text{init}(y : zs) \\
[] ++ ys &\rightarrow ys \\
(x : xs) ++ ys &\rightarrow x : (xs ++ ys) \\
\text{take}(0, xs) &\rightarrow [] \\
\text{take}(s(n), []) &\rightarrow [] \\
\text{take}(s(n), x : xs) &\rightarrow x : \text{take}(n, xs) \\
\text{zip}([], ys) &\rightarrow [] \\
\text{zip}(x : xs, []) &\rightarrow [] \\
\text{zip}(x : xs, y : ys) &\rightarrow (x, y) : \text{zip}(xs, ys)
\end{aligned}$$

Se pide:

- a) Comprobar que \mathcal{R} es inductivamente secuencial, construyendo los árboles definicionales que sean necesarios.
 - b) Aplicando la estrategia NNS explicada en [29], calcular todas las formas posibles de efectuar un pasos de estrechamiento necesario a partir del término $t_1 = \text{take}(n, xs ++ ys)$.
 - c) Repetir el ejercicio del apartado anterior, considerando ahora el término $t_2 = \text{zip}(xs, \text{init}(ys))$.
 - d) Reiterando pasos necesarios en el sentido de la estrategia NNS, construir todas las secuencias de estrechamiento necesario que convierten los términos t_1 y t_2 propuestos en los dos apartados anteriores a forma c-estable.
16. Sea \mathcal{R} el SRT inductivamente secuencial del ejercicio anterior. En cada uno de los dos apartados siguientes se indican un objetivo S para \mathcal{R} y una c-sustitución θ . En ambos casos se pide construir una secuencia de pasos de reescritura necesaria $\theta(S) \rightarrow_{\mathcal{R}^?}^{\text{NR}^*} \underline{t}$, y aplicar la construcción del lema de ascensión de NR a NN (Lema 7.2.6 de [29], pg. 129) para obtener una secuencia de pasos de estrechamiento necesario $S \rightsquigarrow_{\sigma, \mathcal{R}^?}^{\text{NR}^*} \underline{t}$ que calcule una respuesta $\sigma \preceq \theta[\text{var}(S)]$. Hay que recordar que $\mathcal{R}^?$ es el resultado

de añadir a \mathcal{R} las reglas de reescritura inductivamente secuenciales que especifican el comportamiento de $\equiv^?$ como c-convergencia cerrada.

- a) $G \equiv take(n, xs + +ys) \equiv^? y : []$;
 $\theta \equiv \{n \mapsto s(0), xs \mapsto 0 : [], ys \mapsto 0 : [], y \mapsto 0\}$.
- b) $G \equiv zip(xs, init(ys)) \equiv^? (x, y) : []$;
 $\theta \equiv \{xs \mapsto 0 : [], ys \mapsto s(0) : 0 : [], x \mapsto 0, y \mapsto s(0)\}$.

17. Sea \mathcal{R} el SRT inductivamente secuencial formado por las reglas para $+$ y $take$ formuladas en varios ejercicios anteriores.

- a) Estudia el espacio de búsqueda que resulta al intentar resolver el objetivo $n + n \equiv^? m$ utilizando la estrategia de estrechamiento necesario NN y \mathcal{R} como programa.
- b) Tomando como modelo el Ejemplo 7.4.2 de [29] (pg. 137), discute informalmente el comportamiento de LNC_d , $CLNC$ y NN al tratar de resolver cada uno de los objetivos que siguen usando \mathcal{R} como programa.
- 1) $take(s^5(0) + s^5(0), xs) \equiv^? 0 : 0 : 0 : 0 : 0 : 0 : 0 : 0 : 0 : 0 : []$.
2) $take(s^5(0) + s^5(0), xs) \equiv^? ys$.
3) $take(s(n) + s(n), xs) \equiv^? ys$.

Referencias

- [1] S. Antoy. *Definitional trees*. Proc. ALP'92, Springer LNCS 632, pp.143-157, 1992.
(Propuesta original de los árboles definicionales.)
- [2] S. Antoy and M. Hanus. *Compiling Multi-Paradigm Declarative Programs into Prolog*. Proc. FroCoS 2000, Springer LNAI 1794, pp. 171-185, 2000.
(Técnica de implementación del estrechamiento perezoso en Prolog.)
- [3] S. Antoy, R. Echahed and M. Hanus. *A Needed Narrowing Strategy*. Journal of the ACM, 47(4):776-822, 2000.
(Presentación original de la estrategia de estrechamiento necesario en revista.)
- [4] T. Arts and J. Giesl. *Termination of term rewriting systems using dependency pairs*. Theoretical Computer Science, 236:133-178, 2000.
(Presentación original de la técnica de los pares de dependencia en revista.)
- [5] **F. Baader and T. Nipkow**. *Term Rewriting and All That*. Cambridge University Press, 1998.
(Texto de reescritura de nivel introductorio.)

- [6] C. Borralleras and A. Rubio. *Orderings and Constraints: Theory and Practice of Proving Termination*. In *Rewriting, Computation and Proof. Essays Dedicated to Jean-Pierre Jouannaud on the Occasion of His 60th Birthday*. Springer LNCS 4600, 2007.
(Panorámica actualizada de las técnicas de demostración de terminación.)
- [7] N. Derschowitz and J.P. Jouannaud. *Rewrite Systems*. In J. van Leuwen (Ed.), *Handbook of Theoretical Computer Science*, vol. B, ch. 6, Elsevier, pp. 243-320, 1990.
(Artículo clásico sobre reescritura.)
- [8] S. Escobar. *Refining Weakly Outermost-Needed Rewriting and Narrowing*. Proc. PPDP'03, ACM Press, pp. 113-123, 2003.
(Artículo técnico sobre refinamientos del estrechamiento necesario.)
- [9] S. Escobar. *Implementing Natural Rewriting and Narrowing Efficiently*. Proc. FLOPS'04, Springer LNCS 2998, pp. 147-162, 2004.
(Artículo técnico sobre refinamientos del estrechamiento necesario.)
- [10] S. Escobar, J. Meseguer and P. Thati. *Natural Rewriting for General Term Rewriting Systems*. Proc. LOPSTR'04, Springer LNCS 3573, pp. 101-116, 2005.
(Artículo técnico sobre refinamientos del estrechamiento necesario.)
- [11] Francisco J. López-Fraguas, Juan Rodríguez-Hortalá and Jaime Sánchez-Hernández. *A Simple Rewrite Notion for Call-time Choice Semantics*. Proc. PPDP'07, ACM Press, pp. 197-208, 2007.
(Presentación alternativa de la reescritura con semántica CRWL.)
- [12] J.C. González-Moreno, M.T. Hortalá-González, F.J. López-Fraguas and M. Rodríguez-Artalejo. *An Approach to Declarative Programming Based on a Rewriting Logic*. *Journal of Logic Programming*, 40(1):47-87, 1999.
(Presentación original de la reescritura condicional con semántica CRWL en revista.)
- [13] M. Hanus. *The Integration of Functions into Logic Programming: From Theory to Practice*. *Journal of Logic Programming*, 19&20:583-628, 1994.
(Panorámica clásica de técnicas de integración de la programación lógica y la programación funcional.)
- [14] **M. Hanus**. *A Unified Computation Model for Functional and Logic Programming*. Proc. POPL'97, ACM Press, pp. 80-93, 1997.
(Propuesta original de un modelo de cómputo basado en estrechamiento necesario y residuación.)
- [15] **M. Hanus**. *A Unified Computation Model for Declarative Programming*. Proc. APPIA-GULP-PRODE'97, pp. 9-24, 1997.
(Divulgación de un modelo de cómputo basado en estrechamiento necesario y residuación.)

- [16] **M. Hanus.** *Teaching Functional and Logic Programming with a Single Computation Model.* Proc. PLIPL'97, Springer LNCS 1292, pp. 335-350, 1997.
(Uso didáctico de un modelo de cómputo basado en estrechamiento necesario y residuación.)
- [17] M. Hanus. *Multi-paradigm Declarative Languages.* Proc. ICLP'07, Springer LNCS 4670, pp. 45-75, 2007.
(Panorámica actualizada de técnicas de combinación de paradigmas de programación declarativa.)
- [18] N. Hirokawa and A. Middeldorp. *Dependency Pairs Revisited.* Proc. RTA'04, Springer LNCS 3091, pp. 249-268, 2004.
(Presentación actualizada de la técnica de los pares de dependencia en revista.)
- [19] J.M. Hullot. *Canonical Forms and Unification.* Proc. CADE'80, Springer LNCS 87, pp. 318-334, 1980.
(Artículo clásico sobre la técnica de estrechamiento.)
- [20] J.W. Klop. *Term Rewriting Systems.* In S. Abramsky, D.M. Gabbay and T.S.E. Maibaum (Eds.), *Handbook of Logic in Computer Science*, vol. 2, Oxford University Press, pp. 2-116, 1992.
(Artículo clásico sobre reescritura, con énfasis en los sistemas de reescritura ortogonales.)
- [21] M. Marin and A. Middeldorp. *New completeness results for lazy conditional narrowing.* Proc. PPDP'04, ACM Press, pp. 120-131, 2004.
(Resultados técnicos sobre completitud de estrategias perezosas de estrechamiento para sistemas de reescritura condicionales.)
- [22] A. Middeldorp. *Call by Need Computations to Root-Stable Form.* Proc. POPL'97, ACM Press, pp. 94-105, 1997.
(Propuesta original de una estrategia de reescritura orientada al cómputo de formas estables con pasos necesarios.)
- [23] A. Middeldorp and E. Hamoen. *Completeness Results for Basic Narrowing.* *Applicable Algebra in Engineering, Communication and Computing*, 5:213-253, 1994.
(Artículo de revista interesante por su panorámica de resultados sobre completitud de estrategias de estrechamiento.)
- [24] A. Middeldorp, S. Okui and T. Ida. *Lazy narrowing: strong completeness and eager variable elimination.* *Theoretical Computer Science* 167, 95-130, 1996.
(Resultados técnicos sobre completitud de estrategias perezosas de estrechamiento.)

- [25] A. Middeldorp and S. Okui. *A Deterministic Lazy Narrowing Calculus*. Journal of Symbolic Computation, 25(6):733-757, 1998.
(Más resultados técnicos sobre completitud de estrategias perezosas de estrechamiento.)
- [26] E. Ohlebusch. *Advanced Topics in Term Rewriting*. Springer, 2002.
(Texto de reescritura de nivel avanzado.)
- [27] M. Rodríguez-Artalejo. *Functional and Constraint Logic Programming*. In H. Comon, C. Marché and R. Trainen (eds.): *Constraints in Computational Logic, Theory and Applications*. Revised Lectures of the International Summer School CCL'99, Springer LNCS 2002, Chapter 5, pp. 202-270, 2001.
(Panorámica sobre técnicas de integración de la programación funcional y la programación lógica con restricciones.)
- [28] Terese. *Term Rewriting Systems*. Cambridge Tracts in Theoretical Computer Science, 55. Cambridge University Press, 2003.
(Texto de reescritura de carácter enciclopédico.)
- [29] **R. del Vado Vírseda**. *Estrategias de Estrechamiento Perezoso*. Trabajo de Investigación de Tercer Ciclo. DSIP UCM, Septiembre 2002.
(Panorámica de técnicas de estrechamiento perezoso utilizables como mecanismo de cómputo en lenguajes lógico-funcionales.)
- [30] R. del Vado Vírseda. *A Demand-driven Narrowing Calculus with Overlapping Definitional Trees*. Proc. PPDP'03, ACM Press, pp. 213-227, 2003.
(Propuesta de una estrategia de estrechamiento perezoso guiado por árboles definicionales, generalizando la propuesta original de estrechamiento necesario de Antoy, Echahed y Hanus.)