Propuesta de Tesina para la Obtención del Título de Licenciado en Ciencias de la Computación

Guido Martínez

16 de noviembre de 2020

Título de la tesina: ***Estudio de confluencia en sistemas de reescritura probabilistas***

Postulante: Martínez, Guido Omar

Director: Díaz-Caro, Alejandro

1. Situación del Postulante

El postulante tiene aprobadas todas las materias de la carrera (formalmente, falta una optativa) y no se encuentra desempeñando ninguna otra tarea académica ni profesional. Se espera dedicar cerca de 40 horas semanales (*full-time*) a la realización de la tesina.

2. Motivación y Objetivo General

A la hora de dar semántica a un lenguaje, una opción popular es hacerlo operacionalmente. Es decir, definir el *significado* de un programa como la manera en que se ejecuta. Particularmente, en las semánticas *small-step*, los programas se evalúan reduciendo paso a paso hasta, posiblemente, llegar a un valor.

Puede ocurrir que la semántica permita que un programa pueda ejecutarse de dos maneras distintas. En estos casos es importante que las distintas ejecuciones tengan el mismo resultado final. Esta propiedad, apropiadamente formalizada, se conoce como *confluencia*.

Algunos lenguajes de programación (en particular, aquellos que modelan la computación cuántica) tienen semánticas probabilistas, en donde la noción de evaluación lleva una probabilidad asociada. En estos casos, la definición usual de confluencia tiene poco sentido, dado que por lo general sí hay (y se quiere que hayan) caminos que llevan a valores finales distintos.

En estos casos, también es deseable poder reducir un término de distintas maneras (correspondientes a distintas estrategias de reducción), y es importante que estas elecciones no influyan en *el conjunto* de resultados finales.

Si bien hay algunos trabajos al respecto, los estudios de confluencia resultan inaplicables a contextos como la computación cuántica [6], o están acoplados al lenguaje que se estudia [3, 9]. Por lo tanto, pensamos que sería provechoso buscar una definición adecuada de la misma junto a teoremas y condiciones que puedan ser reusables. Un estudio al nivel de los sistemas de reescritura, sin ligarse a un cálculo específico, parece ser una buena opción para esta meta.

3. Fundamentos y Estado del Conocimiento Sobre el Tema

Los lenguajes de programación con reescritura probabilista son utilizados para estudiar diferentes problemas: complejidad (ej. [11]), seguridad (ej. [1]), computación cuántica (ej. [3, 9]), concurrencia (ej. [20]), etc.

Existen muchos lenguajes probabilistas en la literatura, por ejemplo [2, 5, 10, 14, 20, 21, 23–25]. En particular el lenguaje probabilista de Dal Lago y Zorzi [10], fue motivado en agregar medición (una operación intrínsecamente probabilista) al cálculo cuántico de Zorzi [8]. Dentro de los lenguajes cuánticos que utilizan reescritura probabilista, se destaca el cálculo de Selinger y Valiron [26], o el lenguaje Quipper [19]. También podemos destacar los lenguajes no deterministas, como [4,7,12,13,15,17,18,22], ya que es posible convertir fácilmente un lenguaje no determinista en un lenguaje probabilista [16].

Sobre estos lenguajes se han estudiado muchas propiedades. Aunque no hay un estándar aceptado de cómo definir un lenguaje probabilista, un punto en común en la mayoría de los trabajos citados es que el comportamiento probabilista se basa en el sistema de reescritura que define su semántica operacional.

Los lenguajes probabilistas suelen permitir, debidamente, que un término reduzca a dos valores distintos. En este caso, claramente el lenguaje no es confluente en el sentido usual. Este es siempre el caso para los lenguajes con medición cuántica: si tenemos un estado superpuesto, al medirlo obtendremos alguno de los datos en superposición con cierta probabilidad, y claramente esos resultados no serán compatibles. Por este motivo, la mayoría de los lenguajes probabilistas han abandonado por completo el chequeo de confluencia.

Un lenguaje en donde sí se ha estudiado la "confluencia" puede encontrarse en [6]. Sin embargo, la definición usada es muy similar a la tradicional y prohíbe que un programa reduzca a dos valores distintos, lo que hace que no resulte interesante para modelar, por ejemplo, la medición cuántica.

Es necesaria una noción de confluencia que tenga en cuenta la *distribución* de resultados en vez de los resultados particulares. A nuestro saber, sólo dos lenguajes tratan el problema de confluencia de este modo: la extensión con medición al lenguaje cuántico de van Tonder [3] y la extensión con medición al lenguaje cuántico de Zorzi [9]. En ambos casos, se define la confluencia como la confluencia del lenguaje determinista asociado que reescribe conjuntos probabilistas de términos. Adaptar esas pruebas a otros lenguajes no es trivial, ya que cada uno tiene sus particularidades y no demuestran resultados reusables.

Aún siendo un resultado específico del cálculo en cuestión, en [9] se afirma que su análisis es bastante independiente del mismo y se conjetura una condición suficiente para que se logre la confluencia. Una línea clara de trabajo es intentar demostrar o refutar esta conjetura.

Por este motivo, en esta tesina proponemos definir y estudiar la confluencia de sistemas de reescritura probabilistas al nivel de sistemas de reescritura abstractos y de sistemas de reescritura de términos.

4. Objetivos Específicos

Los sistemas de reescritura abstractos y de términos proveen un formalismo matemático en donde estudiar, entre otras cosas, semánticas operacionales de lenguajes de programación. Muchas propiedades de las semánticas de los lenguajes pueden postularse sobre estos sistemas, en donde existen resultados más generales que pueden reusarse, para así obtener una comprensión más profunda de las mismas.

Se propone encarar la confluencia probabilista desde los sistemas abstractos de reescritura, extendidos con reducciones probabilistas. Se espera que se deba definir una noción de sistemas

de reescritura probabilistas (tanto abstractos como de términos) para expresar la propiedad.

Se espera poder desarrollar una noción adecuada de confluencia para lenguajes probabilistas, con aplicación a lenguajes que modelen la computación cuántica; encontrar condiciones necesarias y/o suficientes para que la propiedad se cumpla, simplificando el análisis de cálculos concretos; estudiar las consecuencias de la propiedad y aplicarla a cálculos existentes.

5. Metodología y Plan de Trabajo

Programa tentativo de trabajo:

- Relevar en detalle las distintas definiciones de lenguajes probabilistas existentes: 2 semanas.
- Definir una noción de sistema de reescritura probabilista que englobe a los lenguajes probabilistas de interés y permita diferentes estrategias de reducción: 2 semanas.
- Definir una noción de confluencia sobre los sistemas previos que se corresponda con el hecho de que distintas estrategias de reducción no influyen en la distribución final: 2 semanas.
- Obtener condiciones simplificadas y/o métodos para demostrar la propiedad en lenguajes concretos: 6 semanas.
- Aplicar los resultados a cálculos existentes: 4 semanas.

El trabajo se realizará durante aproximadamente 4 meses con dedicación completa.

6. Referencias Bibliográficas

[1] Musab AlTurki, José Meseguer, and Carl A. Gunter. Probabilistic modeling and analysis of dos protection for the asv protocol. *Electron. Notes Theor. Comput. Sci.*, 234:3–18, March 2009.

[2] S. Andova. Process algebra with probabilistic choice. In J-P. Katoen, editor, *Formal Methods for Real-Time and Probabilistic Systems*, volume 1601 of *LNCS*, pages 111–129, 1999.

[3] P. Arrighi, A. Díaz-Caro, M. Gadella, and J. J. Grattage. Measurements and confluence in quantum lambda calculi with explicit qubits. In B. Coecke, I. Mackie, P. Panangaden, and P. Selinger, editors, *Proceedings of QPL/DCM 2008*, volume 270/1 of *ENTCS*, pages 59–74. Elsevier, 2011.

[4] G. Boudol. Lambda-calculi for (strict) parallel functions. *Information and Computation*, 108(1):51–127, 1994.

[5] O. Bournez and M. Hoyrup. Rewriting logic and probabilities. In R. Nieuwenhuis, editor, *Rewriting Techniques and Applications*, volume 2706 of *LNCS*, pages 61–75, 2003.

[6] O. Bournez and C. Kirchner. Probabilistic rewrite strategies: Applications to ELAN. In Sophie Tison, editor, *Rewriting Techniques and Applications*, volume 2378 of *Lecture Notes in Computer Science*, pages 252–266. Springer-Verlag, 2002.

[7] A. Bucciarelli, T. Ehrhard, and G. Manzonetto. A relational semantics for parallelism and non-determinism in a functional setting. *Annals of Pure and Applied Logic*, 163(7):918–934, 2012.

[8] U. Dal Lago, A. Masini, and M. Zorzi. On a measurement-free quantum lambda calculus with classical control. *Mathematical Structures in Computer Science*, 19:297–335, 2009.

[9] U. Dal Lago, A. Masini, and M. Zorzi. Confluence results for a quantum lambda calculus with measurements. In B. Coecke, P. Panangaden, and P. Selinger, editors, *Proceedings of QPL 2009*, volume 270/2 of *ENTCS*, pages 251–261. Elsevier, 2011.

[10] U. Dal Lago and M. Zorzi. Probabilistic operational semantics for the lambda calculus. *RAIRO - Theoretical Informatics and Applications*, 46:413–450, 2012.

[11] Ugo Dal Lago and Paolo Parisen Toldin. An Higher-Order Characterization of Probabilistic Polynomial Time.

[12] U. de'Liguoro and A. Piperno. Non deterministic extensions of untyped λ-calculus. *Information and Computation*, 122(2):149–177, 1995.

[13] M. Dezani-Ciancaglini, U. de'Liguoro, and A. Piperno. A filter model for concurrent λ-calculus. *SIAM Journal on Computing*, 27(5):1376–1419, 1998.

[14] A. Di Pierro, C. Hankin, and H. Wiklicky. Probabilistic λ-calculus and quantitative program analysis. *Journal of Logic and Computation*, 15(2):159–179, 2005.

[15] A. Díaz-Caro and G. Dowek. Non determinism through type isomorphism. In D. Kesner and P. Viana, editors, *Proceedings of LSFA 2012*, volume 113 of *EPTCS*, pages 137–144, 2013.

[16] A. Díaz-Caro and G. Dowek. The probability of non-confluent systems. In M. Ayala-Rincón, E. Bonelli, and I. Mackie, editors, *Proceedings of DCM 2013*, volume 144 of *EPTCS*, pages 1–15. Open Publishing Association, 2014.

[17] A. Díaz-Caro, G. Manzonetto, and M. Pagani. Call-by-value non-determinism in a linear logic type discipline. In S. Artemov and A. Nerode, editors, *Proceedings of LFCS 2013*, volume 7734 of *LNCS*, pages 164–178, 2013.

[18] A. Díaz-Caro and B. Petit. Linearity in the non-deterministic call-by-value setting. In L. Ong and R. de Queiroz, editors, *Proceedings of WoLLICS 2012*, volume 7456 of *LNCS*, pages 216–231, 2012.

[19] A. S. Green, P. LeFanu Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron. Quipper: a scalable quantum programming language. *ACM SIGPLAN Notices (PLDI'13)*, 48(6):333–342, 2013.

[20] O. M. Herescu and C. Palamidessi. Probabilistic asynchronous π-calculus. In J. Tiuryn, editor, *Proceedings of FoSSaCS 2000*, volume 1784 of *LNCS*, pages 146–160, 2000.

[21] C. Jones and G. Plotkin. A probabilistic powerdomain of evaluations. In *Proceedings of LICS 1989*, pages 186–195. IEEE Press, 1989.

[22] M. Pagani and S. Ronchi Della Rocca. Linearity, non-determinism and solvability. *Fundamental Informaticae*, 103(1–4):173–202, 2010.

[23] S. Park, F. Pfenning, and S. Thrun. A monadic probabilistic language. In *Proceedings of TLDI 2003*, pages 38–49. ACM Press, 2003.

[24] S. Park, F. Pfenning, and S. Thrun. A probabilistic language based upon sampling functions. *SIGPLAN Notices*, 40:171–182, 2005.

[25] N. Ramsey and A. Pfeffer. Stochastic lambda calculus and monads of probability distributions. In *Proceedings of POPL 2002*, pages 154–165. ACM Press, 2002.

[26] P. Selinger and B. Valiron. A lambda calculus for quantum computation with classical control. *Mathematical Structures in Computer Science*, 16(3):527–552, 2006.