Combinando "Model Checking" y "Proof Checking" en el Análisis de Sistemas de Tiempo Real †

Carlos Daniel Luna

Instituto de Computación, Fac. de Ingeniería, U. de la República, Montevideo, Uruguay
E-mail: cluna@fing.edu.uy. Http:www.fing.edu.uy/~cluna
TE: (+598) (2) 7114244 (int. 115), InCo, Uruguay. Fax: (+598) (2) 7110469

Palabras Claves. Especificación y Análisis de Sistemas de Tiempo Real. Autómatas (Grafos) Temporizados. Lógicas TCTL y CTL. Verificación de Modelos ("Model Checking"). Teoría de Tipos y Coq. Verificación-Demostración de Corrección ("Proof Checking").

1. Resumen extendido

Cada vez son más frecuentes las aplicaciones donde el tiempo juega un rol importante. Por ejemplo en: protocolos de comunicación; controladores de robots, de comandos de aviones, de pasos a nivel de trenes, de procesos industriales automatizados y de dispositivos electrónicos (o electro-mecánicos); aplicaciones multimedia y de internet; entre otras. En general éstas son aplicaciones críticas, en las cuales una falla o mal funcionamiento pueden acarrear consecuencias graves, tales como poner en juego vidas humanas y/o grandes inversiones económicas. El comportamiento de estos sistemas, llamados *sistemas de tiempo real*, no está determinado únicamente por la sucesión de acciones que se ejecutan, sino también por el momento en que las mismas ocurren y son procesadas. El tiempo de ejecución es "el" parámetro fundamental en el comportamiento de esta clase de sistemas y una gran parte, quizás la más importante, de los requerimientos de los mismos son temporales: "tal acción debe ejecutarse en un lapso de tiempo determinado", "el tiempo transcurrido entre dos eventos o señales debe estar acotado por un valor constante", etc.

Para el análisis de sistemas reactivos y de tiempo real dos importantes enfoques formales se destacan: la verificación de modelos, o *model checking* [1, 2, 3, 4, 5, 6, 7, 8], y el análisis deductivo basado en asistentes de pruebas (por ejemplo, [9, 10, 11, 12, 13]). El primer enfoque se caracteriza por la automaticidad pero presenta dificultades al tratar con sistemas que involucran un gran número de estados o donde se tienen parámetros variables, no acotados. El segundo permite tratar con sistemas arbitrarios pero requiere la interacción del usuario.

Esta investigación explora una metodología de trabajo que permita compatibilizar el uso de un verificador de modelos como *Kronos* [14] y el asistente de pruebas *Coq* –basado en una teoría constructiva de tipos– [15] en el análisis de sistemas de tiempo real. Para ello se formalizan *grafos* (*autómatas*) *temporizados* [2, 16, 17] y la lógica *TCTL* [2, 3, 18] (y su restricción *CTL* [1, 19] para sistemas reactivos) en el cálculo de construcciones inductivas [20] y co-inductivas [21, 22] de *Coq*, a fin de disponer de lenguajes de especificación y análisis comunes a ambas herramientas. Los grafos permiten describir los sistemas, mientras que la lógica se usa para especificar requerimientos

† Una versión preliminar de este artículo es el reporte [28] (http://www.fing.edu.uy/~cluna).

temporales. *Kronos* permite verificar automáticamente si un grafo temporizado satisface una fórmula *TCTL*, siempre que los parámetros del sistema sean valores constantes. El análisis deductivo permite trabajar con parámetros variables y de esta manera verificar sistemas más generales. En este contexto, a las ventajas tradicionales[1] del enfoque deductivo respecto de la verificación automática se suma una muy importante: la posibilidad de realizar síntesis de programas a partir de una formalización en teoría de tipos. Esta es una línea que no es explotada en el presente trabajo pero que justifica aún más el interés de esta experiencia.

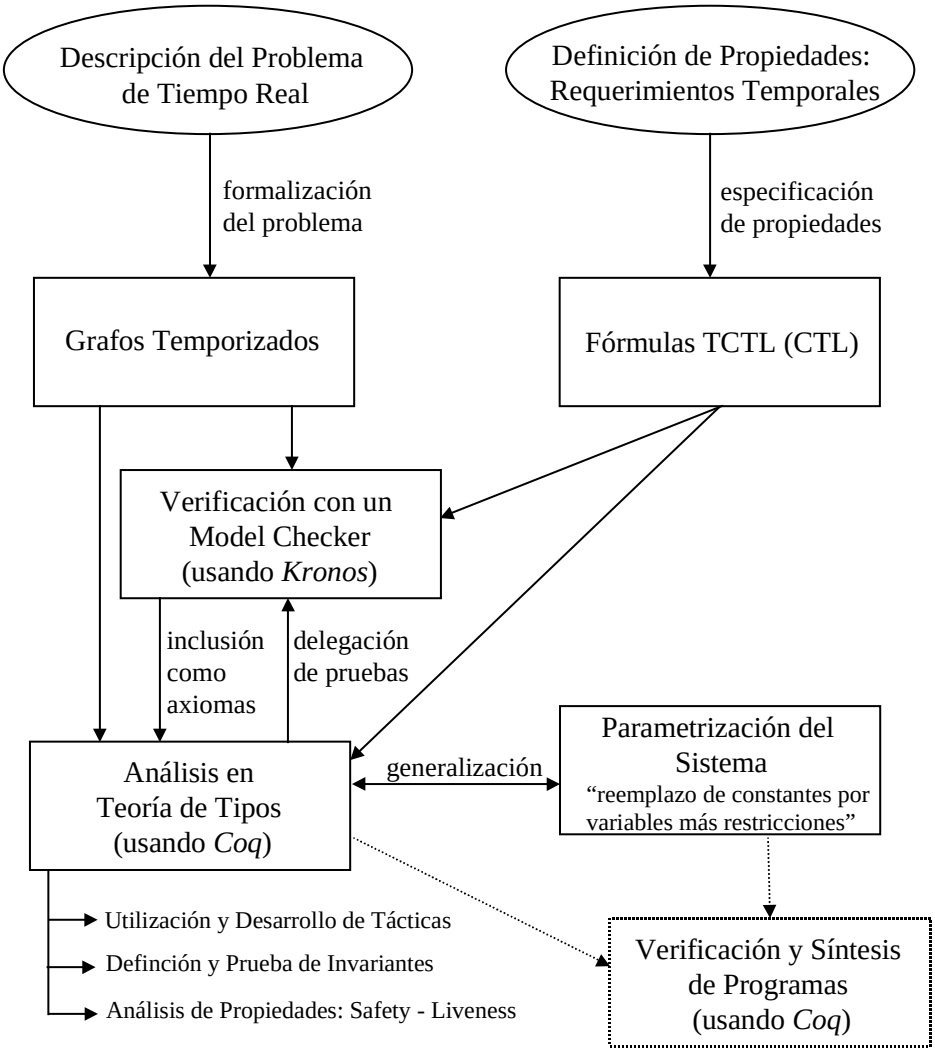
Una parte importante del trabajo está dedicada a estudiar cómo razonar deductivamente en *Coq* sobre sistemas de tiempo real, evaluando la utilidad de tipos inductivos y la necesidad de tipos co-inductivos, asumiendo inicialmente un modelo de tiempo discreto. Se incluyen algunas definiciones alternativas para los operadores que permiten expresar invarianza y alcanzabilidad, con tipos inductivos (bajo la noción de estados alcanzables) y co-inductivos (bajo la noción de trazas –infinitas– de ejecución). Estos últimos necesarios para una descripción completa de todas las fórmulas de las lógicas TCTL y CTL. Asimismo, se formulan y demuestran en *Coq* propiedades generales de los operadores temporales que permiten simplificar ciertas pruebas (particularmente, propiedades de invarianza y alcanzabilidad).

El trabajo define dos representaciones genéricas de grafos temporizados para la semántica de tiempo discreto considerada, una de las cuales es extendida a tiempo continuo. Estas representaciones permiten obtener sistemas como instancias particulares y simplifican el proceso de definición de sistemas compuestos con el uso de un operador genérico de composición (evitando definir la composición de dos o más grafos temporizados de forma particular cada vez).

Un especial énfasis es puesto en el análisis de un caso de estudio, considerado como benchmark en diferentes trabajos: *el control de un paso a nivel de tren* (“*the railroad crossing example*” [3, 11, 13, 16, 23, 24, 25, 26, 27]). Este problema es utilizado para evaluar y validar algunas de las formalizaciones propuestas [29]. A partir del análisis de un conjunto de invariantes del sistema –que expresan propiedades cualitativas y cuantitativas temporales del sistema– se prueba la propiedad de seguridad esencial del mismo (*safety*): “si un tren está cruzando el paso a nivel, la barrera está baja” y la propiedad de *liveness Non-Zeno* que justifica la corrección temporal del sistema. Las demostraciones son parametrizadas en un conjunto de restricciones entre los parámetros del sistema, inicialmente considerados constantes y posteriormente generalizados a variables sujetas a un conjunto de restricciones establecidas –deducidas a partir de las condiciones de prueba de las propiedades que se decidieron preservar. De esta manera se generaliza la especificación del sistema original considerado [16], que es tratable con un *model checker* sólo para valores constantes de los parámetros.

[1] Ver, por ejemplo, [28].

2. Esquema de la metodología de trabajo



Referencias

1. E. Clarke, E. Emerson, and A. Sistla. "Automatic verification of finite-state concurrent systems using temporal logic specifications". *ACM Transactions on Programming Languages and Systems*, 8(2):244-263, 1986.
2. R. Alur, C. Courcoubetis, and D. Dill. "Model-checking for real-time systems". In *Proc. 5th Symp on Logics in Computer Science*, pages 414-425. IEEE Computer Society Press, 1990.
3. T. Henzinger, X. Nicollin, J. Sifakis, and S. Yovine. "Symbolic model-checking for real-time systems". In *Proc. 7th Symp on Logics in Computer Science*. IEEE Computer Society Press, 1992.
4. R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Oliver, J. Sifakis, and S. Yovine. "The algorithmic analysis of hybrid systems". *Theorical Computer Science*, 138:3-34, 1995.
5. F. Laroussinie and K. Larsen. "Compositional model checking of real time systems". In *Proc. 6th Conference on Concurrency Theory*, pages 27-41, Philadelphia, Springer-Verlag, 1995.

6. O. Sokolsky and S. Smolka. “Local model checking for real-time systems. In *Computer Aided Verification, Proc. 7th Int. Workshop*, Liege, *LNCS* 939, pages 211-224, 1995.
7. T. Henzinger, O Kupferman, and M. Vardi. “A space-efficient on-the-fly algorithm for real-time model checking”. In *Proc 7th Conference on Concurrency Theory*, *LNCS* 1119, pages 514-529, Springer-Verlag, 1996.
8. H. Saïdi and N. Shankar. “Abstract and model Check while you prove”. In *CAV’99*, Trento, Italy, 1999.
9. S. Owre, J. Rushby, and N. Shankar. “PVS: A prototype verification system”. In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE)*. *LNIA* 607, Saratoga, NY, 1992. Springer Verlag.
10. M. Gordon. *Introduction to HOL: a theorem proving environment based for higher order logic*. Cambridge University, Press, 1993.
11. N. Shankar. “Verification of real-time systems using PVS”. *CAV’93*, Costas Courcoubetis, editor, Elounda, Greece, *LNCS* 697, pages 280-291, 1993.
12. L. Paulson. “The Isabelle reference manual”. *Technical Report* 283, Computer Laboratory, University, 1993.
13. N. Bjørner, Z. Manna, H. Spima, and T. Uribe. “Deductive Verification of Real-time Systems Using SteP”. *ARTS-97*, vol. 1231 of LNCS, pp. 22-43, Springer-Verlag, 1997.
14. S. Yovine. “Kronos: A verification tool for real-time systems”. *Software Tools for Technology Transfer*, 1997.
15. B. Barras, S. Boutin, C. Cornes, J. Courant, Y. Coscoy, D. Delahaye, D. de Rauglaudre, J-C. Filliâtre, E. Giménez, H. Herbelin, G. Huet, H. Laulhère, C. Muñoz, Ch. Murthy, C. Parent-Vigouroux, P. Loiseleur, Ch. Paulin-Mohring, A. Saïbi, and B. Werner. “The Coq Proof Assistant. Reference Manual, Versión 6.2.4”. *INRIA*, 1999.
16. R. Alur and D. Dill. “A theory of timed automata”. *Theorical Computer Science*, 126:183-235, 1994.
17. A. Olivero. *Modélisation et Analyse de Systèmes Temporisés et Hybrides*. PhD thesis, Institut National Polytéchnique de Grenoble. France, 1994.
18. R. Alur. *Techniques for automatic verification of real-time systems*. PhD thesis, Departament of Computer Science, Stanford University, 1991.
19. E. Amerson and E. Clarke. “Using branching-time temporal logic to synthesize synchronization skeletons”. *Science of Computer Programming*, 2:241-266,1982.
20. C. Paulin-Mohring. “Inductive definitions in the system Coq – rules and properties”. In M. Bezem and J. Groote, editors, *Proceeedings of the conference Typed Lambda Calculi and Aplications*, *LNCS* 664, 1993.
21. L. Paulson. “Co-induction and Co-recursion in Higher-order Logic”. *Technical Report* 304, Computer Laboratory, University of Cambridge, 1993.
22. E. Giménez. *A Calculus of Infinite Constructions and its application to the verification of communicating systems*. PhD thesis, Ecole Normale Supérieure de Lyon, 1996, Unité de Recherche Associée au CNRS No. 1398, 1996.
23. Z. Chaochen, C. Hoare, and A. Ravn. “A calculus of durations”. *Information Processing Letters*, 40(5):269-276,1992.

24. C. Heitmeyer, R. Jeffords, and B. Labaw. “A benchmark for comparing different approaches for specifying real-time systems”. *Real Time: Theory and Practice, LNCS* 600, REX Workshop, Mook, The Netherlands, 1991. Springer-Verlag.

25. T. Henzinger and O. Kopke. “Verification methods for the divergent runs of clock systems”. In *FTRTFT’94: Formal Techniques in Real-time and Fault-tolerant Systems*, volume 863 of *LNCS*, pages 351-372. Springer-Verlag, 1994.

26. C. Daws and S. Yovine. “Verification of multirate timed automata with KRONOS: two exemples”. Technical Report Spectre-95-06, VERIMAG, 1995.

27. J. Armstrong and L. Barroca. “Specification and verification of reactive systems behaviour: The railroad crossing example”. *Real-Time Systems*, 10:143-178, 1996.

28. C. Luna. “Especificación y análisis de sistemas de tiempo real en teoría de tipos. Caso de estudio: *the railroad crossing example*”. Technical Report 00-01, InCo, PEDECIBA Informática, Fac. de Ingeniería, U. de la República, Uruguay, 2000.

29. C. Luna. “Verificación de Sistemas de Tiempo Real en Teoría de Tipos. Un Caso de Estudio: The RailRoad Crossing example in Coq”. En proceedings de la *Conferencia Latinoamericana de Informática: CLEI'2002*, Montevideo, Noviembre de 2002.