Ana Gabriela Garis

Lógica Temporal en Verificación de Modelos de Software. Origen y Evolución hasta tiempos actuales

Fundamentos en Humanidades, vol. XI, núm. 1, 2010, pp. 151-161,
Universidad Nacional de San Luis
Argentina

Disponible en: http://www.redalyc.org/articulo.oa?id=18415426010

Fundamentos en Humanidades,
ISSN (Versión impresa): 1515-4467
fundamen@unsl.edu.ar
Universidad Nacional de San Luis
Argentina

Fundamentos en Humanidades
Universidad Nacional de San Luis – Argentina
Año XI – Número I (21/2010) 151/161 pp.

Lógica Temporal en Verificación de Modelos de Software. Origen y Evolución hasta tiempos actuales

Temporal Logic in the Verification of Software Models. Origin and Evolution up to current times

Ana Gabriela Garis
Universidad Nacional de San Luis
agaris@unsl.edu.ar

(Recibido: 06/03/09 – Aceptado: 12/05/09)

Resumen

En sistemas de software críticos, tales como sistemas de control de vuelo de aviones o sistemas de control de ascensores, los errores de software deben ser evitados. En la actualidad, existen técnicas y herramientas para asegurar la correctitud de sistemas críticos, siendo "model checking" (chequeo de modelos) una de las más populares. Con model checking, tanto el sistema de software como las propiedades deseadas del sistema (seguridad, alcanzabilidad, etc.) se modelan utilizando lenguajes de especificación formal; es decir, lenguajes basados en conceptos lógico-matemáticos. Las propiedades son especificadas normalmente a través de un tipo de lógica denominada "lógica temporal". Profundizando sobre dicha lógica, el presente trabajo muestra los aspectos epistemológicos relacionados, detallando sobre su origen y evolución, desde las nociones de lógicas introducidas por Aristóteles hasta su concepción como herramienta clave para la verificación de sistemas de software críticos.

Abstract

In critical software systems, such as control systems for aircrafts or elevators, software errors should be avoided. Nowadays, there exist techniques and tools for ensuring correctness in critical systems, being "model checking" one of the most popular. With model checking, software systems

and desired system properties (security, reachability, etc.) are modeled using formal specification languages, i.e. specification languages based on logical-mathematic concepts. Properties are often specified using a type of logic called “temporal logic”. Going into such logic, the present work shows the related epistemological features, detailing the origin and evolution of temporal logic from the first concepts about logics introduced by Aristotle to its conception as a key tool for verifying critical software systems.

Palabras clave

ingeniería de software - verificación de modelos - chequeo de modelos - lógica temporal

Key words

software engineering - model verification - model checking - temporal logic

Introducción

La Ingeniería de software es una disciplina de la informática referida a la aplicación práctica del conocimiento científico para el diseño y construcción de programas de computadoras, junto con la documentación requerida para su desarrollo, operación y mantenimiento (Boehm, 1976). El proceso de desarrollo de software comprende una serie de fases; tales como la especificación de los requerimientos del usuario, el análisis de los requerimientos, el diseño de soluciones y la implementación, prueba y mantenimiento del software implementado. En las diferentes etapas normalmente se elaboran modelos que permiten razonar sobre el problema a resolver.

Un modelo de software es una descripción del sistema y su ambiente, el cual es utilizado con diferentes propósitos. Los modelos ayudan a los desarrolladores de SW a analizar el problema que se desea resolver y a comunicarse con otros ingenieros, al tiempo que facilitan la abstracción de los aspectos relevantes y la tarea de reutilización de las diferentes piezas de software. En la actualidad, la importancia que presentan los modelos del sistema hace que se dedique tiempo y esfuerzo en procura de obtener ‘buenos’ modelos; a tal punto que se han propuestos metodologías de desarrollo de software dirigidas completamente por los modelos; tales como MDA (Object Management Group, 2003). La especificación de modelos tiene una influencia crítica

en la calidad del software final, disminuyendo notablemente la cantidad de errores. Se invierte tiempo y dinero en el testeo y debugging de programas para la detección temprana de errores, ya que el costo de fallas en un sistemas implica pérdidas cuantiosas. Un estudio del año 2002 estimó que el costo económico de los errores en programación en Estados Unidos asciende a los $ 60 Billones anuales (National Institute of Standards and Technology, 2002). Mas aún, existen sistemas críticos; tales como el sistema que maneja un ascensor, o sistemas de manejo de aviones, donde el error no es tolerable, dado que implicaría posiblemente la pérdida de vidas humanas.

Las técnicas de verificación de software ofrecen un medio para evaluar la calidad de sistemas de software (Preece, 1998). La verificación de un modelo establece si su lógica ha sido correctamente implementada; es decir, se refiere a la construcción del sistema correcto (Boehm, 1984). Existen diferentes métodos de verificación dependiendo del tipo de sistema a chequear. El método de verificación empírica propone la simulación controlada del software para detectar posibles fallas. Pero en sistemas complejos donde no hay lugar para el error, los métodos formales son más adecuados. Los métodos formales utilizan un enfoque matemático tanto para la especificación, diseño y verificación de sistemas de software.

Model checking (MC) es en la actualidad una de las técnicas mas populares de verificación formal. El modelo del sistema se especifica por un lado; y por otro lado, se escriben propiedades tales como seguridad, usando el tipo de lógica conocida como “lógica temporal”. Otro tipo de lógicas también son utilizadas dependiendo del tipo de sistema a chequear. Por ejemplo, en sistemas de agentes inteligentes utilizados en dominios de aplicación como bibliotecas digitales o mercados virtuales, se utiliza lógica epistémica.

La lógica temporal ha sido utilizada como estructura para definir semántica de expresiones temporales en los más diversos dominios; en el presente trabajo se presenta como herramienta para manejar aspectos temporales en la ejecución de programas de computación. En particular, se profundiza sobre el origen y la evolución de la lógica temporal hasta la actualidad.

La siguiente sección describe la técnica de model checking y lógica temporal, detallando sobre las características principales de model checking, el significado de algunas propiedades deseadas del sistema a ser especificadas con lógica temporal y la noción de operadores temporales. Posteriormente, se detalla sobre su origen y evolución, mostrando los fundamentos brindados por las personas más destacadas a lo largo de su historia. Finalmente se exponen las conclusiones del trabajo.

Model Checking y Lógica temporal

Model checking es una técnica automática para la verificación de sistemas finitos. Habilitar MC frecuentemente requiere la construcción de un modelo simplificado más pequeño, más abstracto que el diseño principal, preservando características esenciales pero evitando complejidad. La verificación utilizando model checking exige la especificación del sistema y de las propiedades deseadas. Algunas de las propiedades típicas que suelen ser especificadas son:

- Alcanzabilidad: es posible alcanzar una situación.
- Seguridad: es seguro que una situación no sucederá.
- Vivacidad: si empieza debe terminar.
- Equidad: un número infinito de veces una situación se podrá dar, se dará o no se dará nunca.

Las propiedades son especificadas normalmente a través de lógica temporal. El proceso MC consiste en la especificación de un modelo, especificación de propiedades y verificación. Esto significa que dada una propiedad deseada *p* y una estructura del modelo *M* con un estado inicial *s*; decide si *p* cumple *M*; o más precisamente, decide si *M, s* |= *p*.

El modelo *M* es normalmente expresado como un sistema de transición, un grafo dirigido que conforma un árbol computacional (una estructura Kripke). Cada nodo tiene asociado un conjunto de proposiciones atómicas. Los nodos representan estados del sistema, los arcos posibles transiciones y las proposiciones atómicas propiedades básicas que se cumplen en un punto de ejecución. MC está basado en el cálculo y representación de todos los estados posibles alcanzables por el sistema. Una herramienta de model checking típica responde 'si', si el modelo satisface las especificaciones que establecen las propiedades; en caso contrario, genera un contra-ejemplo. De esta forma los chequeadores de modelos son usados como debuggers para encontrar problemas, como así también para mostrar la correctitud una vez que se ha finalizado el debugging.

El comportamiento deseado del sistema se especifica con un lenguaje formal, generalmente usando lógica temporal. Con lógica temporal la noción estática de verdad es reemplazada por una dinámica, en la cual las fórmulas pueden cambiar sus valores de verdad a medida que el sistema pasa de un estado a otro.

La lógica temporal extiende a la lógica clásica permitiendo especificar en qué momento del tiempo esta teniendo lugar un hecho o proposición. Por ejemplo, en la lógica clásica dos enunciados tales como "sale el sol" y "saldrá el sol" deben ser especificados como proposiciones diferentes, o como la

misma proposición. En lógica temporal, se utiliza una sola proposición y se distingue diferentes estados del tiempo utilizando operadores temporales. De esta forma, se pierde la funcionalidad de verdad; por ejemplo, en el caso anterior, tendríamos la sentencia "sale el sol" independiente del tiempo, y su valor de verdad cambiaría dependiendo del sistema.

La lógica temporal se encuentra dentro de las llamadas lógicas no-clásicas. Una interesante clasificación de lógicas no-clásicas, se encuentra planteada en (Muñoz Gutierrez, 2000), donde se divide aquellas lógicas que son extensiones de la lógica clásica; tales como la lógica modal, de otras que son alternativas de la lógica clásica: las lógicas trivalentes, lógica intuicionista, lógicas multivalentes, lógicas de la probabilidad, lógicas borrosas, lógicas nomonotónicas. Dentro de la lógica modal existen diferentes interpretaciones: la lógica temporal, la lógica epistémica y la lógica deónica.

Existen diferentes formalismos de lógica temporal; tales como Linear-time temporal logic (LTL), Computational tree logic (CTL) y CTL*. Las fórmulas CTL* están compuestas por cuantificadores de ramas o paso y operadores temporales. Los cuantificadores de paso describen las ramificaciones del árbol computacional. Desde un estado particular, permiten especificar que todos los pasos o algunos de los pasos comenzando desde un estado tiene/n alguna propiedad. Informalmente se define a los cuantificadores de paso de la siguiente manera:

- A: para todos los pasos o ramificaciones del árbol computacional
- E: para alguno de los pasos o ramificaciones del árbol computacional

Los operadores temporales describen propiedades de un paso o rama del árbol computacional. En la figura 1 es posible observar la semántica de alguno de los operadores temporales; los cuales son definidos informalmente de la siguiente manera:

- X: próximo momento del tiempo (next)
- F: eventualmente o en el futuro (eventually)
- G: siempre en el futuro (always)
- U: hasta (until)
- R: hasta el momento de liberación (release)

Los operadores temporales son utilizados en conjunto con los operadores lógicos tradicionales (not, and, or), para definir las propiedades deseadas del sistema. Por ejemplo, las propiedades de seguridad, que establecen que no es posible alcanzar un estado peligroso particular, tienen la forma *AG not s*, donde "*s*" es un estado peligroso de riesgo. Una fórmula especificada con LTL se representa como una computación sobre una secuencia infinita de estados. Las fórmulas LTL no utilizan el

cuantificador existencial, ya que presuponen que la fórmula debe cumplirse para todos las ramas del árbol computacional. Las lógicas CTL y CTL* proveen cuantificadores explícitos (A y E).

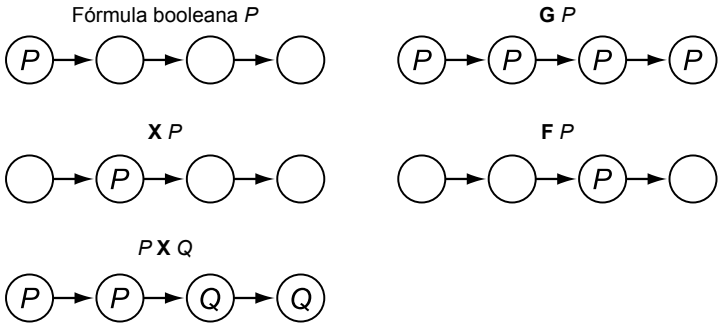

Figura 1: Semántica intuitiva de algunos operadores temporales

Lógica temporal: Origen y Evolución

Comenzando con Aristóteles (384 - 322 a. C.), la lógica ha sido ampliamente utilizada en la filosofía para clarificar conceptos tales como la causalidad, la necesidad histórica, la identidad a lo largo del tiempo y las nociones de sucesos y acciones (Vázquez Campos, 2001). Aristóteles es el primero en formalizar expresiones para analizar inferencias entre enunciados, construye el primer sistema de lógica de términos, introduce el concepto de cuantificación de enunciados y define el silogismo lógico.

> "El silogismo es un discurso en el cual, puestas ciertas cosas, algo distinto de las cosas puestas se sigue necesariamente de ellas, como consecuencia suya, y sin que sea preciso introducir ningún otro término para justificar la necesidad de la conclusión" (Aristóteles, Analíticos primeros, I, 24b).

Aristóteles establece una relación entre los valores de verdad (verdad o falsedad) de premisas y conclusiones, y la corrección o incorrección del razonamiento lógico (Klimosky, 1994). El ejemplo clásico de silogismo lógico donde se muestra la forma del razonamiento y los valores de verdades de premisas y conclusiones, es el siguiente:

Todos los hombres son mortales
Todos los griegos son hombres
Todos los griegos son mortales

Los dos primeros enunciados son las premisas y el tercero, “Todos los griegos son mortales”, es la conclusión. El ejemplo expone un caso donde tanto las premisas como la conclusión son verdaderas y reconoce la forma que un razonamiento debe tener para ser considerado correcto:

Todo B es C
Todo A es B
Todo A es C

Los términos genéricos A, B y C representan sustantivos o adjetivos del lenguaje natural. Los términos pueden ser reemplazados por cualquier sustantivo o adjetivo, pero para que el razonamiento sea válido éstos deben ocupar siempre la misma posición dentro del enunciado. Esto garantiza que cualquier caso particular donde se encuentre un razonamiento con dicha forma, si las premisas son verdaderas, la conclusión necesariamente también lo será.

La lógica medieval, desarrollada durante la Edad Media, del siglo XI al XV, hereda de la lógica griega la silogística aristotélica. Arthur Prior (1914 - 1969) destaca las aportaciones de la Escolástica mencionando al desarrollo de la lógica de las modalidades como uno de los aspectos fundamentales. La lógica modal define diferentes modalidades de verdad, distinguiendo si una proposición P, es verdadera en circunstancias presentes, posiblemente (*possibly*) sea verdadera bajo algunas circunstancias, o si necesariamente (*necessarily*) P se cumpla bajo todas las circunstancias.

La lógica modal, introducida por Prior, estudia la relación entre tiempo y modalidad de Diodoro Cronos (340 - 280 a.C.). La innovación en lógica más notable de Diodoro está referida al “problema de las contingencias futuras” relacionada a qué cosas son posibles. El mismo problema fue re actualizado más tarde por Aristóteles y posteriormente discutido por Leibniz (1646-1716).

Aristóteles distingue entre oraciones atributivas y oraciones modales. En las oraciones atributivas solo se afirma o se niega que el predicado P sea o no atribuible al sujeto S; en las modales se indica además la manera en cómo P se relaciona con S. Define a las oraciones modales como aquellas constituidas por dos elementos *modus* y *dictum*, y determinando

que el modo no afecta a un componente de la proposición sino a toda. Por ejemplo, en la oración "es posible que salga el sol", el *modus* es "es posible" y el *dictum* es "salir el sol". Las modalidades impuestas por Aristóteles son cuatro: necesidad, imposibilidad, posibilidad y contingencia.

Tiempo más tarde, Gottfried Leibniz piensa un lenguaje simbólico universal como instrumento del cálculo del pensamiento; por lo cual propone varios cálculos lógicos. El aporte de Leibniz a la lógica modal se relaciona a la interpretación del lenguaje modal definido por la semántica del mundo racional o posible. El concepto de necesidad de éste Leibniz es la siguiente: Una proposición es mirada para ser necesariamente si y solo si no es solo verdadera en el mundo real, sino verdadera en todos los mundos posibles.

Esto está directamente asociado al concepto de estructura Kripke que posibilita la creación de un árbol computacional. Una estructura Kripke se define como un par K=<W,R>, donde W es un conjunto no vacío de mundos y R una relación binaria arbitraria de W. Así, la semántica para los operadores de necesidad (G) y posibilidad (F) para una proposición modal *p* es como se sigue: *Gp* será verdadera en un mundo *x* si es verdadera en todos los mundos alternativos a *x*; y *Fp* será verdadera en un mundo *x* si es verdadera en al menos un mundo *y* tal que *y* es una alternativa a *x* (Chagrov y Zakharyaschev, 1997).

La evolución de la lógica modal según Orayen (Orayen, 1995), se divide en cuatro etapas: prehistoria (desde Aristóteles hasta 1912), sintaxis (1912-1959), semántica (1959-1977) y metalógica (1977-actualidad). De la primera etapa se ha desarrollado en párrafos anteriores, en los próximos párrafos se describirá las siguientes 3 etapas.

En la etapa de la sintaxis, Clarence Lewis (1883 - 1964), da origen a la lógica modal moderna y a la lógica polivalente. En 1918 publica "A Survey of Symbolic Logic" en donde propone un nuevo condicional, diferente al de la lógica clásica. En el condicional lógico tradicional, los patrones de razonamiento válidos indican una relación dada por una consecuencia lógica (normalmente denotado por '->') entre enunciados condicionales y enunciados consecuentes. Sin embargo, este tipo de razonamiento trae conclusiones extrañas si se recoge el significado del condicional del lenguaje natural. Lewis introduce un nuevo tipo de condicional denominado "implicación estricta", agregando a la implicación clásica la noción de necesidad (denotado como □).

Los protagonistas más relevantes de la etapa semántica son Saul Kripke y Jaakko Hintikka. Kripke establece semántica a la lógica modal. En su libro "Semantical Considerations on Modal Logic" (consideraciones semánticas

sobre la lógica modal), publicado en 1963, Kripke responde al enfoque clásico que utiliza términos para referirse a cosas que existen sólo contingentemente. La semántica Kripke asigna un valor de verdad relativo a cada punto de un modelo, donde cada punto representa un mundo posible. Los modelos Kripke han sido adoptados como tipo estándar para las lógicas no clásicas relacionadas en general, y para la lógica modal en particular. Los elementos de los modelos Kripke pueden representar diversos significados: momentos del tiempo, situaciones evidentes o estados de una computadora. Hintikka creó una semántica para la lógica modal similar a la de Kripke y fundó la lógica epistémica como subcampo de la lógica modal. El objetivo de la lógica epistémica es razonar sobre el conocimiento, Hintikka sugiere utilizar modalidades para capturar la semántica del conocimiento.

Luego de la etapa semántica, comienza la etapa metalógica con el trabajo de John Lemmon (1930-1966) y Dana Scott, publicado en 1977, "An introduction to modal logic" (Una introducción a la lógica modal). Scott y Lemmon inician la creación del libro pero su desarrollo se interrumpe con el fallecimiento de Lemmon. Scott pasa los manuscritos parciales a sus colegas, lo que permite introducir importantes técnicas en la semántica de la teoría de modelos.

La lógica modal en general y la lógica temporal en particular han sido utilizadas en las más diversas áreas; que van desde la filosofía hasta la ciencias de la computación, pasando por lengua y matemática. Arthur Prior y Rod Burstall son dos de los autores más relevantes en lo que respecta a la utilización de lógica modal en sistemas computacionales. Prior (1967) y Burstall (1974) proponen lógica temporal como adecuada para razonar sobre sistemas de software; pero el trabajo de Amir Pnueli es quizás el que más contribuye a la introducción de la lógica temporal para la verificación de programas y sistemas (Emerson, 2008).

Tal como fue mostrado en párrafos anteriores, Prior estudia la relación entre tiempo y lógica, inspirándose en los trabajos de Diodoro Cronos referidos a la definición de implicación. Supone que es posible relacionar las ideas de Diodoro con trabajos contemporáneos sobre modalidad, desarrollando un cálculo formal el cual incluya operadores temporales análogos a los operadores de la lógica modal. Prior propone una interpretación temporal, un conjunto de posibles mundos ordenados puede corresponder a una secuencia temporal de estados. De esta manera, los operadores modales "*necessarily*" y "*possible*" se transforman en los cuantificadores temporales "*always*" y "*eventually*" y se introducen los operadores "*next*" y "*until*" basados en la lineabilidad del tiempo.

De acuerdo a Sain, el problema original que Prior y Burstall atacan son dos:

1). “Encontrar caracterizaciones del poder de verificación de programas de diferentes métodos de verificación de programas reconocidos, en términos de lógica clásica de primer orden. Tales caracterizaciones deben ser simples y matemáticamente transparentes.

2). Comparar el poder de la verificación de programas de diferentes métodos reconocidos” (1985: 1).

Burstall entiende que es necesario clarificar la naturaleza de las lógicas especiales para razonar sobre programas y propone a la lógica temporal en ciencias de la computación por primera vez, enfoque mejorado mas tarde por Pnueli en su trabajo titulado “The Temporal Logic of Programs” (La lógica temporal de Programas) (Pnueli, 1977). En dicho trabajo, Pnueli presenta un enfoque para la verificación de programas concurrentes (programas que tienen más de una linea lógica de ejecución) a partir de un método de prueba que utiliza el razonamiento temporal.

A partir de Pnueli hasta la actualidad se ha seguido utilizando la lógica temporal para la verificación formal de sistemas de software, pero también a cobrado importancia en otras áreas de la Ciencias de la Computación como en Inteligencia Artificial. El éxito de las lógicas no-clásicas, tanto en Ciencias de la Computación como en Ingeniería de Software, ha permitido la introducción y evolución en los más diversos dominios, tanto de lógicas modales (temporal, epistémica, deónica), como de otras lógicas tales como las trivalentes y probabilísticas.

Conclusiones

El presente trabajo ha mostrado el origen y la evolución de la lógica temporal desde las primeras nociones brindadas por Aristóteles hasta su concepción como herramienta para la verificación de sistemas computacionales. El marco histórico ha mostrado, cómo a medida que model checking ha crecido en popularidad, se ha incrementado la importancia de la lógica temporal dentro de las Ciencias de la Computación y de la Ingeniería de Software. En la actualidad, dependiendo del tipo de sistema a chequear, se aplican otras lógicas no-clásicas, entre las que se encuentra, entre otras, la lógica espistémica.

San Luis, 4 de marzo de 2009

Referencias bibliográficas

Aristóteles (1988). Analíticos primeros. En M. de Candel Sanmartín (trad. esp.) *Tratados de lógica (Órganon) II. Sobre la interpretación. Analíticos primeros. Analíticos segundos*. Madrid: Gredos.

Boehm, B. (1976). Software Engineering. *IEEE Transactions on Computers*, C-25, Nº 12, 1226–1241.

Boehm, B. (1984). Verifying and validating software requirements and design specifications*, IEEE Software, Vol. 1(1).*

Burstall, R. M. (1974). Program Proving as Hand Simulation with a Little Induction*, IFIP Congress,* pp. 308-312.

Chagrov, A. y Zakharyaschev, M. (1997). *Modal Logic.* Clarendon Press.

Emerson, E. A. (2008). The Beginning of Model Checking: A Personal Perspective. *Lecture Notes In Computer Science, 25 Years of Model Checking: History, Achievements, Perspectives, pp 27-45, Springer-Verlag.*

Klimosky, G. (1994). *Las desventuras del conocimiento científico.* Buenos Aires: A-Z Editoral.

Muñoz Gutierrez, C. (2000). Lógicas No-Clásicas. Apunte de Cátedra, Lógica y Computación, Departamento de Lógica y Filosofía de la Ciencia, Universidad Complutense de Madrid, España.

National Institute of Standards and Technology (2002). *Software Errors Cost U.S. Economy $59.5 Billion Annually*. US Department of Commerce, NIST News Release.

Object Management Group (2003). *Model Driven Architecture Guide*. Disponible en http://www.omg.org/mda

Orayen R., Alchourrón C. y Méndez J. (1995). Lógica. Vol. 7, Enciclopedia Iberoamericana de Filosofía, Madrid.

Preece A. (1998). Building the Right System Right. *AAAI-98 Workshop on Verification and Validation of Knowledge-Based Systems, Technical Report WS-98-11, AAAI Press.*

Prior A. (1967). *Past, Present, and Future*. Oxford University Press.

Pnueli, A. (1977). The Temporal Logic of Programs. *Proceedings of the 18th IEEE Symposium on Foundations of Computer Science, pp. 46-57.*

Sain, I. (1985). The Reasoning Powers of Burstall's (modal logic) and Pnueli's (Temporal Logic) Program Verification Methods*, Lecture Notes In Computer Science; Vol. 193, pp. 302 - 342 , Springer-Verlang.*

Vázquez Campos, M. (2001). Breve Introducción a la Lógica Temporal. *Revista Laguna*, pp. 187-198, España.