PONTIFICIA UNIVERSIDAD CATOLICA DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIA DE LA COMPUTACION

Tópicos en Ciencia de la Computación - IIC3800
Verificación Formal de Software
Programa de Curso

Horario cátedra	:	Martes y Jueves, módulo 3, sala Javier Pinto
Profesores	:	Marcelo Arenas (`marenas@ing.puc.cl`)
		Pablo Barceló (`pbarcelo@dcc.uchile.cl`)
URL	:	`www.ing.puc.cl/~marenas/iic3800`

Objetivo

La verificación formal es una técnica que permite verificar sistemas concurrentes con un número finito de estados, tales como circuitos secuenciales o protocolos de comunicación. Esta técnica tiene varias ventajas sobre otras técnicas tradicionales basadas en la simulación, prueba, y razonamiento deductivo. En particular, la verificación formal es automática y eficiente. Además, si el diseño contiene un error, la verificación formal puede ser usada para identificarlo.

El método de verificación formal ha sido exitosamente utilizado en la práctica para verificar diseños industriales, y varias empresas han empezado a comercializar sistemas de verificación. Además, su inventor, Amir Pnueli, obtuvo el Turing award en 1996 por la introducción de este método a la Ciencia de la Computación.

Este curso tiene como fin introducir al alumno a las técnicas básicas de verificación formal de software y sistemas concurrentes. Se presentarán, entre otras, las técnicas de *modelación*, de *especificación* usando lógicas temporales (LTL, CTL, etc), y de *verificación* usando autómatas. Se estudiarán también las técnicas usadas para tratar el problema de la *explosión de estados*, es decir, cuando el número de estados en el sistema crece exponencialmente con el número de procesos que interactúan. Al final del curso se dará énfasis a la implementación de estos métodos en verificadores aplicados en la industria, como por ejemplo SPIN, SMV, etc.

Metodología

El curso se basa en clases expositivas de 80 mins. cada una. Se realizará un promedio de 2 clases semanales.

Evaluación

Se realizarán seis controles. La nota final será el promedio de los cinco mejores controles.

Contenido

1. Motivación.
2. Modelación de software usando autómatas.
3. Lógicas temporales: LTL, CTL, CTL*. Expresividad de lógicas temporales.
4. Algoritmo de verificación para CTL.
5. Autómatas sobre palabras infinitas. Autómatas alternadores. Equivalencia de ambos modelos.
6. Algoritmo de verificación para LTL. Algoritmo de verificación para CTL*.
7. Diagramas de decisión binaria. Verificación simbólica.
8. μ-calculus. Expresividad con respecto a otras lógicas temporales. Algoritmos de verificación para el μ-calculus.
9. Abstracción. Reducción del cono de influencia. Abstracción de datos.
10. Verificador SPIN. Ejemplos. Técnicas utilizadas por SPIN.

Bibliografía

1. Transparencias de clases.
2. E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.
3. B. Berard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, Ph. Schnoebelen, and P. McKenzie. *Systems and Software Verification: Model-Checking Techniques and Tools*. Springer-Verlag, 2001.