Model Checking

Сергей Геннадьевич Синица
КубГУ, 2013
sin@kubsu.ru

Учебный план

Лекции 12 часов:

- Моделирование программ и систем, введение в SPIN
- Спецификация проверяемых свойств, LTL
- Разбор примеров верификации программ
- Разбор примеров верификации параллельных программ
- Обзор и сравнение программ для ModelChecking с примерами (NASA/JPL, SMV, Uppaal, Kronos)
- Алгоритмы верификации и оптимизации

Практика 26 часов

Домашняя работа 100+ часов

Работа, баллы и оценки

Практическая работа и индивидуальное задание 60 баллов

- учебные задания из документации SPIN 15
http://spinroot.com/spin/Man/Exercises.html
- индивидуальная задача на SPIN 15
- задачи на LTL15
- доклад 15

Экзамен письменно (?) 40 баллов

Пропущенная пара -2 балла

Оценка - Баллы

5 >= 90

4 >= 75

3 >= 60

Темы докладов

Алгоритм Model Checking для CTL, пример
Алгоритм Model Checking для LTL, пример
Бинарные решающие диаграммы, свойства, операции, применение
Символьный алгоритм Model Checking для CTL, пример
Алгоритм Model Checking для PCTL, пример
Алгоритм Model Checking для TCTL, пример
SPIN optimization: partial order reduction
SPIN optimization: bitstate hashing
SPIN optimization: minimised automaton representation of reachable states
SPIN optimization: state vector compression
SPIN optimization: dataflow analysis and slicing algorithm
Обзор и сравнение программ для ModelChecking: NASA/JPL (для Java кода)
Обзор и сравнение программ для ModelChecking: SMV (символьный)
Обзор и сравнение программ для ModelChecking: Uppaal (в реальном времени)
Обзор и сравнение программ для ModelChecking: Kronos (в реальном времени)

Источники

Spin Model Checker manual and examples.
URL: http://spinroot.com/spin/Man

Ю.Г. Карпов. Model Checking. Верификация параллельных и распределенных программ

При разработке программы использованы материалы:

Spin Workshops, Advanced Spin Tutorial, Theo C. Ruys & Gerard J. Holzmann
Курс «Верификация моделей программ», лектор Владимир Захаров
Курс «Верификация программ на моделях» лектор Константин Савенков
LTL Spec Patterns, SAnToS laboratory
Учебное пособие «Верификация программ методом Model Checking» А.М.Миронов
Concise Promela Reference, Rob Gerth
Книга «Верификация моделей программ», Кларк, Грамберг, Пелед
Курс «Logic Model Checking», CS118, Caltech, Gerard J. Holzmann

Model Checking

Clarke & Emerson 1981:

"Model checking is an automated technique that, given a finite-state model of a system and a logical property, systematically checks whether this property holds for (a given initial state in) that model."

Принципы Model Checking

1. Для заданной вычислительной системы (программы или проекта микроэлектронной схемы) ***построить модель M***
   - дискретную структуру, которая
   - может рассматриваться как интерпретация для некоторой формальной логики, и
   - описывает (на некотором уровне абстракции) поведение вычислительной системы.
2. Для технических требований, предъявляемых к системе, сформулировать эти требования на формальном логическом языке ***задать спецификацию ф***.
3. Проверить выполнимость спецификации ф на модели M:

M |= ф

Model Checking

Темпоральные логики позволяют описывать порядок событий во времени:

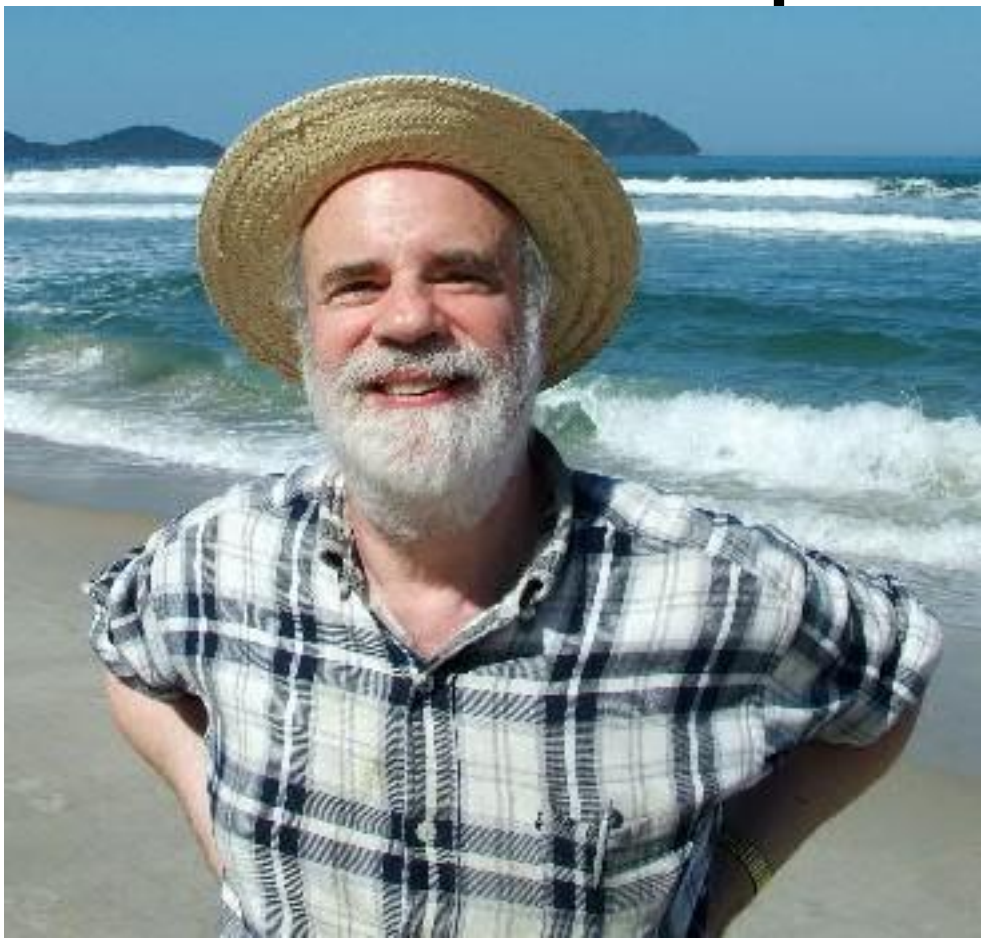
- CTL (Computational Tree Logic, ветвящаяся структура времени)
- LTL (Linear Temporal Logic, линейная)
- PCTL (Probabilistic, вероятностная)
- TCTL (в реальном времени)

Смысл всякой формулы темпоральной логики определяется по отношению к размеченному графу переходов.

Исторически структуры такого вида называются **моделями Крипке (Kripke structure)**.

Задача model checking **M |= ϕ**

Saul Aaron Kripke



Родился 13 ноября 1940 г.
Американский философ и логик
Семантика Крипке для модальной логики
Naming and Necessity (Prinston, 1972, 1980)

Модель Крипке

Рассмотрим множество атомарных высказываний AP.

Моделью Крипке M над множеством AP назовем четверку **M = (S, S0 , R, L)**:

1) S — конечное множество состояний;
2) So $\subseteq$ S — множество начальных состояний;
3) R $\subseteq$ S × S — отношение переходов, которое обязано быть тотальным, т. е. для каждого состояния s $\in$ S должно существовать такое состояние s $\in$ S , что имеет место R(s, s);
4) L : S $\rightarrow$ 2^{AP} — функция разметки, которая помечает каждое состояние множеством атомарных высказываний, истинных в этом состоянии.

Пример

(Волк, Коза, Капуста, Лодочник)

AP ={ЛодочникСлева, КозаЖива, КапустаЦела}

So = (←, ←, ←, ←)
S = {(←, ←, ←, ←), (←, →, ←, →), (←, †, →, →), …}
R = {((←, ←, ←, ←), (←, →, ←, →)),
((←, ←, ←, ←), (←, †, →, →)), …}

L((←, ←, ←, ←)) = (ЛодочникСлева, КозаЖива,
КапустаЦела)
L((←, †, →, →)) = (КапустаЦела)

Пример

Рассмотрим программу с переменными
x, y ∈ D, D = {0, 1}.

Программа содержит один переход
x := (x + y)(mod2),

В начальном состоянии
x = 1 и y = 1.

Пример

Программа будет охарактеризована двумя формулами первого порядка:

$So(x, y) \equiv x = 1 \ \& \ y = 1,$

$R(x, y, x, y) \equiv x = (x + y)(mod2) \ \& \ y = y.$

Модель Крипке $M = (S, S0, R, L)$ такова:

$S = D \times D;$

$So = \{(1, 1)\};$

$R = \{((1, 1), (0, 1)), ((0, 1), (1, 1)),$
$((1, 0), (1, 0)), ((0, 0), (0, 0))\};$

$L((1, 1)) = \{x = 1, y = 1\}, L((0, 1)) = \{x = 0, y = 1\},$
$L((1, 0)) = \{x = 1, y = 0\}, L((0, 0)) = \{x = 0, y = 0\}.$

Единственный путь в модели Крипке, исходящий из начального состояния, имеет вид (1, 1)(0, 1)(1, 1)(0, 1) . . .

Модель Крипке на практике

Степень детализации (сразу ли умрет коза)

Параллельное исполнение (синхронное и асинхронное, взаимодействие через общую память и сообщения)

Трансляция программ в модели Крипке (методы оптимизации по памяти)

Model Checking на практике

Водопадная модель [Pressman 1996]:

1) System Engineerig
2) Analysis
3) Design
4) Code
5) Testing
6) Maintenance

Классический и современный Model Checking выполняется до и после кодирования соответственно