A Markov Reward Model for Software Reliability

YoungMin Kwon and Gul Agha
Open Systems Laboratory
Department of Computer Science
University of Illinois at Urbana-Champaign
Motivation

- Evaluating software with many processes/threads
  - State explosion problem
    - 100 3-state system => $3^{100}$ states
- Statistical abstraction of state
  - State ($pmf$): probability that a randomly picked thread is in certain module.
    - e.g.: 90% of the threads are in ‘A’ state
  - Abstract enough to handle state
  - Detailed enough to evaluate reliability
Overview

- Model the software as Markov reward model
  - Each module represents a state
  - Module reliabilities are *rewards*
  - Transition probabilities between modules are obtained by *operational profiling*

- System Reliability Estimation
  - Evaluated by *Probabilistic Model Checking*
  - Helps focus testing on particular modules that may increase the reliability of the entire system
    - Modules are not equally important
Markov Reward Model for Software Reliability

Markov model

- Model the program by a DTMC $X = (S, M)$
  - $S$ is the set modules in the program and $M$ represents the transition probabilities between modules.

- Reliability of a module:
  - Probability that a module does not produce a fault when a control is passed to it.
  - NHPP: a simple reliability model
    \[
    R(x | t) = e^{-a} \left( e^{-b t} - e^{-b(t+x)} \right)
    \]
    $R(x|t)$ is the probability that a module does not have a failure during the time interval $t$ to $t + x$. 
Markov model

DTMC model extension for reliability checking

- Add fail state \( f \).
- Transition probability matrix \( M \) is extended as follows

\[
M' = \begin{bmatrix}
    r_1 M_{11} & r_2 M_{12} & \cdots & r_n M_{1n} & 0 \\
    r_1 M_{21} & r_2 M_{22} & \cdots & r_n M_{2n} & 0 \\
    \vdots & \vdots & \ddots & \vdots & \vdots \\
    r_1 M_{n1} & r_2 M_{n2} & \cdots & r_n M_{nn} & 0 \\
    1 - r_1 & 1 - r_2 & \cdots & 1 - r_n & 1
\end{bmatrix}.
\]
Reliability

Reliability of a program is the probability that a program eventually arrives at the final success state:

\[
r = \mathbf{M}_{n^*} \lim_{t \to \infty} \sum_{i=1}^{t} \mathbf{M}_*^i \cdot \mathbf{x}(0)
\]

where \( \mathbf{M}_* \) is a sub-matrix of \( \mathbf{M} \) that comprises the first \( n - 1 \) rows and the first \( n - 1 \) columns of \( \mathbf{M} \), \( \mathbf{M}_{n^*} \) is a \( n \)th row vector of \( \mathbf{M} \) with first \( n - 1 \) elements, and \( \mathbf{x}(0) \) is an initial probability mass function of \( X(0) \).
Example

A module reliability diagram, where $r_A$, $r_B$, and $r_C$ are the reliabilities of the modules A, B, and C.

Reliability of the program is modeled by a DTMC $X = (S, M)$ where $S = \{A, B, C, \text{success, fail}\}$ and $M$ as above.
Example…

Reliability of a program as a function of module Reliabilities \( r_A \) and \( r_B \) with \( r_c = 1 - 10^{-5} \).

Transition probabilities significantly affect the overall reliability of the program.
Probabilistic Model Checking

- With PCTL like logics, ‘P[X=A] > .3 is always true’ is always false regardless of initial pmfs
- However if 50 out of 100 threads are in A state and the others are in B state ⇒ 50% of the threads are always in A state.
- iLTL can specify this situation because it works on pmf transitions
iLTL Formula

Syntax

\[ \psi ::= T \mid F \mid ineq \mid \\
    \neg \psi \mid \psi \lor \phi \mid \psi \land \phi \mid \\
    X\psi \mid \psi U \phi \mid \psi R \phi \]

\[ ineq ::= \sum_{i=1}^{n} a_i \cdot P[X = s_i] < b \]
iLTL Formula
: atomic propositions

- At least 10% more nodes are in READY state than in IDLE state
  - $P[X=\text{READY}] > P[X=\text{IDLE}] + 0.1$
  - $P[X=\text{READY}] - P[X=\text{IDLE}] > 0.1$

- Expected Queue length is less than 2
  - $1*P[Q=\text{S1}] + 2*P[Q=\text{S2}] + 3*P[Q=\text{S3}] < 2$

- Availability of a system X is 10% larger than that of a system Y
  - $P[X=\text{READY}] > P[Y=\text{READY}] + 0.1$
Main Theorem

- If
  - Markov matrix $M$ is diagonalizable
  - Absolute value of second largest \textit{eigenvalue} of $M$ is strictly less than 1
  - For all inequalities of an iLTL formula $\Psi$, the steady state expected value of LHS is not equal to its RHS

- Then
  - There is a bound $N$ after which all inequalities of $\Psi$ become constants
Model Checking Algorithm

Markov model & iLTL specification

Markov model, inequalities
compute search depth

iLTL
build a Buchi automaton

check feasibility through LP

YES
NO with a counterexample
iLTL Model Checking of Software Reliability

Program properties related to reliability that can be evaluated using iLTL are

- Find the configuration of a system (represented by pmf) that will make the system most unreliable.
- Reliability of a system given its configuration.
- Effects on the reliability of the program if different executions constraints are enforced on the program.
- System parameter adjustment through comparison between systems with different parameters.
Model not appropriate….

- iLTL model checking algorithm cannot be directly applied on the previous Markov model because the model violated the eigenvalue constraints of the theorem.
- Transformation is required.
Add Success State

- Fail state is replaced by done state and the self loop transition of success state is removed.
- Transition from success to done with a probability one is added and success state is made transient.
Modification…

- Modified DTMC model is $X = (S,M)$ where $S = \{ A, B, C, \text{success}, \text{done} \}$ and

$$M = \begin{bmatrix}
.95r_A & .07r_B & .05r_C & .00 & .00 \\
.05r_A & .90r_B & .00r_C & .00 & .00 \\
.00 & .02r_B & .95r_C & .00 & .00 \\
.00 & .01r_B & .00 & .00 & .00 \\
1-r_A & 1-r_B & 1-r_C & 1.0 & 1.0
\end{bmatrix}.$$  

- The reliability of the program is the accumulated sum of the probabilities that the success state is visited. It is given by

$$r = \sum_{t=0}^{\infty} P\{X(t) = \text{success}\}$$
$$= \sum_{t=0}^{\infty} [0, 0, 0, 1, 0] \cdot x(t)$$
$$= \sum_{t=0}^{\infty} [0, 0, 0, 1, 0] \cdot M^t \cdot x(0).$$
Figure shows how the probabilities of each states change over time and how the reliability of the program \( P\{X(t) = \text{success}\} \) is accumulated with the module reliabilities \( r_A, r_B \) and \( r_C \) and initial pmf \( x(0) = [1/3, 1/3, 1/3, 0, 0] \)
model:
   Markov chain pgm
   has states:
      { A, B, C, S, D},
   transits by:
      [ .9215, .0699, .05, 0, 0; 
       .0485, .8991, 0, 0, 0; 
       0, .02, .9191, 0, 0; 
       0, .01, .03, 0, 0; 
       .03, .001, .001, 1.0, 1.0 ]

specification:
   a : .2149*P{pgm=A} + .3478*P{pgm=B}  + .5036*P{pgm=C} < .7,
   b : .2149*P{pgm=A} + .3478*P{pgm=B} + .5036*P{pgm=C} < .5,
   c : .2149*P{pgm=A} + .3478*P{pgm=B} + .5036*P{pgm=C} < .3,
   d : P{pgm=S} + P{pgm=D} > .0,
   e : P{pgm=A} > P{pgm=C} + .3

   a                     # 1)
   #(b ∧ ¬ d) -> ¬ e   2)
   #(b ∧ ¬ d) -> <>¬ e  3)

   An iltl checker description of the modified reliability model
Results

- The specification “a” checks whether the reliability of the program pgm is less than 0.7
  
  Depth 22
  Result T

  Here 22 indicates the required search depth for the formula.

- The second example (b → ~e) checks whether the fact that the reliability of pgm is less than 0.5 implies not e
  
  Depth 78
  Result F

  Counterexample: \text{pmf} (\text{pgm}(0)) : [.3 \ 0 \ 0 \ 0 \ 0]

  Result is false and it provides with the counter example of X(0).
Figure explain the second and third examples. From step 1 to 15, the probability difference is larger than 0.3. However, eventually after step 15 the difference becomes less than .3.
Future Work

- Develop a distributed algorithm for iLTL to speed up model checking
  - iLTL model checking is a feasibility checking of Disjunctive Normal Form of (in)equality constraints: each conjunctive set of constraints can be checked independently.
  - Once search depth N is computed, bounded model checking techniques can be introduced.