Quantitative Modeling
in
Maude

Carolyn Talcott
SRI International
MQLA 2009
PLAN

• RWL and Maude
• Reasoning about Time: RealTime Maude
• Probablistic modeling: PMaude
• XTune
What is Rewriting Logic

• A logic for executable specification and analysis of concurrent, distributed and/or mobile systems
• A logic to specify other logics or languages
• An extension of equational logic with local rewrite rules expressing
  • concurrent change over time
  • inference rules
Rewrite Theories

- Rewrite theory: $(\text{Signature}, \text{RewriteRules})$
- Signature: $(\text{Sorts}, \text{Ops}, \text{Equations})$ -- an equational theory describing system state
- Rewrite rule: $label: t \rightarrow t' \text{ if } \text{cond}$
- Rewriting operates modulo equations
- Generates computations / deductions
Deduction Rules

one step rewrite: \[ \text{reflexivity:} \quad \text{congruence:} \quad \text{replacement:} \]

closed under
Maude

http://maude.cs.uiuc.edu

- Maude is a language and tool based on RWL
- High performance rewriting modulo axioms
- Modularity, builtins, reflection
- Execution, search, model checking
Petri Net Model of a Vending Machine

mod VENDING-MACHINE is
  sorts Coin Item Place Marking .
  subsorts Coin Item < Place < Marking .
  op null : -> Marking .
    *** empty marking
  ops $ q : -> Coin .
  ops a c : -> Item .
  op _ _ : Marking Marking -> Marking
    [assoc comm id: null] .
    *** multiset
  rl[buy-c]: $ => c .
  rl[buy-a]: $ => a q .
  rl[change]: q q q q => $ .
endm
Execution and search

What is one way to use 3 $s?

Maude> rew $ $ $ .
result Marking: q a c c

How can I get 2 apples with 3 $s?

Maude> search $ $ $ =>! a a M:Marking .

Solution 1 (state 8)
M:Marking --> q q c

Solution 2 (state 9)
M:Marking --> q q q a

No more solutions.
states: 10  rewrites: 12)
Model checking

Starting with 5 $s$, can we get 6 apples without accumulating more than 4 quarters?

\[
\text{eq } \forall m(M) \rightarrow n\text{Apples}(n) = \text{countPlace}(M,a) = n .
\]

\[
\text{eq } \forall m(M) \rightarrow \text{lte4Q} = \text{countPlace}(M,q) \leq 4 .
\]

Maude> red modelCheck(vm(5 $s$s$s$s$s),
[~(lte4Q U nApples(6))].
result ModelCheckResult: counterexample(...)

Is value conserved?

Maude> red modelCheck(vm(5 $s$s$s$s$s),[~val(20)].
result Bool: true
Real Time Rewrite Theories
&
RealTime Maude
Real Time Rewrite Theory (RTRwT)

- RT = ((S,O), E, R, φ, τ)
- (((S,O),E),R) is an ordinary Rewrite Theory
- φ interprets an abstract notion of time
- τ maps rules to terms of sort Time
  - τ(l) > 0 -- a tick rule,
  - τ(l) = 0 -- instantaneous rule
- R -- l: t => t’ in time τ(l) if cond
- Computations/derivations: RT |= t -r-> t’
  - each step instantiates rule, picks a time
  - r is the sum of the times of individual steps
Clock example

R, R' range over Time, \( \tau_{running} = R' \) ...

crl[running]:

\{clock(R)\} => \{clock(R + R')\} in time R' if R' <= 24 monus R

rl[reset]: \{clock(24)\} => \{clock(0)\}

rl[batterydies]: \{clock(24)\} => \{stopped-clock(24)\}

rl[stopped]:

\{stopped-clock(R)\} => \{stopped-clock(R + R')\} in time R'
Analysis

- Property logic: rtLTL
  - propositional LTL without Next
  - propositions may refer to time
- Analyses [possibly time bounded]
  - execution
  - search
  - model checking
Sampling

To execute, a strategy is needed to pick times

- Transform RT to $RT^{maxDef(r)}$ (mte sampling)
  - time picked is max allowed by rule condition
  - $r$ is used for the max for unbounded rules

Completeness for mte sampling

$RT, t_0 |= \Phi$ iff $RT^{maxDef(r)}, t_0 |= \Phi$

if RT is time-robust, atoms of $\Phi$ are tick-invariant
tick rule form: \( \text{conf} \Rightarrow \text{delta} (\text{conf}, R') \text{ in time } R' \text{ if } R' \leq \text{mte} (\text{conf}) \)

Clock ticks:

\( \text{crl[running]}: \{\text{clock}(R)\} \Rightarrow \{\text{clock}(R + R')\} \text{ in time } R' \text{ if } R' <= 24 \text{ monus } R \)

\( \text{rl[stopped]}: \{\text{stopped-clock}(R)\} \Rightarrow \{\text{clock}(R + R')\} \text{ in time } R' \)

For running and stopped: \( \text{delta}(\text{clock}(R), R') = \{\text{clock}(R + R')\} \)

For running: \( \text{mte}(\text{clock}(R)) = 24 \text{ monus } R \)

For stopped: \( \text{mte}(\text{clock}(R)) = \text{INF} \)

There are simple conditions on delta and mte that guarantee time-robustness.

Frequently properties are tick-invariant because they don’t mention variables/attributes changed by delta.
Clock analyses

(tsearch $[1]$ \{clock(0)\} $\Rightarrow^*$ \{clock(X:Time)\}
such that $X$:Time $> 24$ in time $\leq 99$.)

$eq \{stopped-clock(R)\} \models clock-dead = true.$
$eq \{clock(R)\} \models clock-is(R') = (R == R').$
$eq \{clock(R)\} \text{ in time } R' \models \text{clockEqualsTime} = (R == R').$

(mc \{clock(0)\} $\models^t$ clockEqualsTime $U$
\text{clock-is(24)} $\lor$ clock-dead) \text{ in time } $\leq 1000$.)
Example analyses

- AER/NCA suite of protocols for reliable, scalable, and TCP-friendly multicast in active networks -- correctness, performance (worst case times).

- OGDC (Optimal Geographical Density Control) wireless sensor network algorithm for picking active nodes
  - Always reach stable/sensing state
  - bound on time to stable state, coverage
  - Wide-mouth frog key sharing -- search for matching connections, attacks
Probabilistic Rewriting & Maude
Probabilistic Rewrite Theory

- $PR = ((S,O), E, R, \pi)$
- $((S,O), E, R)$ is a rewrite theory
- $\pi$ maps rules to probability distribution functions
- $\text{prl } l : t(x) \Rightarrow t'(x,y)$ if $C(x,y)$ with probability $y := \pi_l(x)$

Probabilistic Rewriting Temporal Logic

- $Pq \#p \varphi \quad -- \ q \in \{\forall, \exists\}, \ # \in \{\leq, \geq, <, >\}$
- probability that $\varphi$ holds on all/some paths is $\# p$
Expressiveness

Diagram showing relationships between tools and formalisms:
- PNS
- GSMP
- PRISM
- CTMC
- PMaude
- FPRTh
- PRwTh
- Tools (GMSim, etc.)
- R_{PNS}, R_{GSMP}, R_{CTMC}
- PNS_R, GSMP_R
- CTMC_R
pml: clock(t,c) ⇒
    if B then clock(t+1, c - c/1000 ) else broken(t, c - c/1000 ) fi
    with probability B := BERNOULLI(c/1000) .

crl: clock(t,c) ⇒
    if B then clock(t+1, c - c/1000 ) else broken(t, c - c/1000 ) fi
    if B := float(random(seed)/maxRand) < c/1000) .
Analysis methods

• testing -- Monte Carlo simulation
• statistical model checking -- Vesta tool
• CSL properties
• statistical qualitative analysis: Quatex language
• $E[\text{term}]$ with error bound, confidence
Analyzing TCP/IP SYN Attack

- Problem: attacker fills syn-queue
- Counter measure -- only check fraction $p$ of syn’s (client must sent multiple requests)
- Analysis: (for different $p$ )
  - expected number of (of 100) clients that successfully connect
  - probability that client connects within time $t$ of initiating a request
  - probability of successful attack $\leq .01$
Statistical MC

- Cache size = 10,000
- timeout = 10 seconds
- number of valid senders = 100

<table>
<thead>
<tr>
<th>Model-checking $\mathbb{P}_{\leq 0.01}(\Diamond (\text{successful_attack}()))$</th>
<th>$X$'s attack rate (SYNs per second)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p = 0.0$ (No counter-measure)</td>
<td>1</td>
</tr>
<tr>
<td>result time ($10^2$ sec)</td>
<td>F</td>
</tr>
<tr>
<td>$p = 0.9$ (With counter-measure)</td>
<td>47</td>
</tr>
<tr>
<td>result time ($10^2$ sec)</td>
<td>F</td>
</tr>
</tbody>
</table>
Quatex Analysis

Expected number of clients out of 100 clients that get connected with the server under DoS attack
XTune
Cross layer adaptive tuning
A. Formal Executable Specification

System Specification:
layered modeling with cross-layer adaptation

Observer/Property Checker:
extract properties/values from executable specification

B. Controller

Policy/Parameter Selection

Observables
(i.e., properties, values)

Simulated execution
(i.e., dynamic system execution behavior)

C. System Realization

Task/OS Module:
application, scheduling

Device Module:
hardware features

Environment Module:
mobility, network status
XTune approach

- System components/layers modeled as objects
- Rules mix time and probability
  - combine ideas of RTMaude and PMaude
- Analysis simplifies/improves ideas of PMaude
Example: Mobile Multimedia

Multi-mode Multimedia Terminal System

User Interface -> FSM Control -> Connection Handling

Video Phone, VOD Player, SMS, Email Client, MP3 Player

AVI Reader -> H.263 Decoder -> Display
       |            |            |
       VOD Player| Email Client| MP3 Player |
       G.723 Encoder

MP3 Player Mode

File Reader -> MP3 Decoder -> Speaker

Video Phone Mode

Camera -> H.263 Encoder
         |            |
         Mux       |            |
         Network   |            |
         Demux     |            |
         G.723 Decoder |            |
         Speaker

G.723 Encoder
XTune model of video phone

System state -- a clocked configuration

\[\{ < \text{CPU: HW} | \text{Timer: 0, policy: P, consumedEnergy: 0.0, ...} > \\
< \text{pbpair: Application} | \text{Timer: 0, accEncTime: 0, consecutiveMiss: 0, ...} > \\
< \text{Mobility: NetworkMonitor} | \text{Timer: 0, pos: L, speed: 1, ...} > \\
< \text{ZoneInfo: Zone} | \text{currentDLY: dly, currentPLR: alpha ...} > \\
< \text{Random: RandomNGen} | \text{seed: N} > \\
... \} \text{ in time 999999} .\]

crl [tick]: \{\text{conf} in time T } \Rightarrow \{\text{delta(conf,T')}\} \text{ in time (T minus T')} \\
\text{if } T' := \text{mte(conf) } \land \ T > T' \land T' > 0 .

Application execution times, packet arrival times ... sampled from 
normal and exponential distributions.
Experiments: Statistical MC

Quick detection of problematic situations (e.g., battery expires)
Sequential testing
  Property \([\text{probability (battery expires)} < 0.1]\)
Parameters
  \(\alpha \) (false negative) = 0.05, \(\beta \) (false positive) = 0.05
  \(\theta \) (threshold) = 0.1, \(\delta \) (indifference region) = 0.01
133 traces give H1 accept

Black-box testing also confirms the formula
with error of 8.20E-7 with same traces.

Performance
  The run time for each statistical model checking is 10-20 msecs
  in addition to the sample generation
  a feasible proposition for the on-the-fly adaptation
Experiments: Statistical Analysis

(a) Energy Consumption:
   \[ n_{\text{Sample}} = 100 \] Fail to reject Ho (p-value = 0.821)
   \[ E[\text{Energy Consumption}] = 3.7121\times10^9 \ (\alpha = 5.0\%, \ d = 0.036\%) \]

(b) Decoder Average Deadline Miss Ratio:
   \[ n_{\text{Sample}} = 100 \] Reject Ho (p-value = 0.035)
   \[ n_{\text{Sample}} = 110 \] Fail to reject Ho (p-value = 0.194)
   \[ E[\text{Decoder Avg Deadline Miss Ratio}] = 0.2032 \ (\alpha = 5.0\%, \ d = 0.466\%) \]

(c) Decoder Maximum Consecutive Lost:
   \[ n_{\text{Sample}} = 100 \] Fail to reject Ho (p-value = 0.884)
   \[ n_{\text{Sample}} = 100 \] (\( d = 0.01053 \) > (\( \delta = 0.01 \))
   \[ n_{\text{Sample}} = 110 \] (\( d = 0.01002 \) > (\( \delta = 0.01 \))
   \[ n_{\text{Sample}} = 121 \] (\( d = 0.00958 \) ≤ (\( \delta = 0.01 \))
   \[ E[\text{Decoder Maximum Consecutive Lost}] = 3.2314 \ (\alpha = 5.0\%, \ d = 0.958\%) \]

(b) The first normality (JB) test fails. Need more samples.
(c) The confidence interval from initial samples is greater than the desired interval.
   \implies\text{Need more samples}
Summary

• Quantitative analysis in Maude is done by
  • extending basic rewriting with time and probabilities (a built in random number generator)
  • mapping special syntax to core Maude
  • execution, search, and various forms of model checking / statistical analysis
References

Maude


RTMaude


References

PMaude


XTUNE