Probabilistic Model Checking
Overview

◆ Crowds redux

◆ Probabilistic model checking
  • PRISM model checker
  • PCTL logic
  • Analyzing Crowds with PRISM

◆ Probabilistic contract signing
  • Rabin’s beacon protocol
  • Ben-Or, Goldreich, Rivest, Micali protocol
  • Analyzing probabilistic contract signing protocols with PRISM
- Routers form a random path when establishing connection
  - In onion routing, random path is chosen in advance by sender
- After receiving a message, honest router flips a biased coin
  - With probability $P_f$ randomly selects next router and forwards msg
  - With probability $1-P_f$ sends directly to the recipient
Probabilistic Model Checking

- Participants are finite-state machines
  - Same as Mur_φ
- State transitions are probabilistic
  - Transitions in Mur_φ are nondeterministic
- Standard intruder model
  - Same as Mur_φ: model cryptography with abstract data types
- Mur_φ question:
  - Is bad state reachable?
- Probabilistic model checking question:
  - What’s the probability of reaching bad state?
Discrete-Time Markov Chains

\[(S, s_0, T, L)\]

- \(S\) is a finite set of states
- \(s_0 \in S\) is an initial state
- \(T : S \times S \rightarrow [0,1]\) is the transition relation
  - \(\forall s, s' \in S \quad \sum_{s'} T(s, s') = 1\)
- \(L\) is a labeling function
Markov Chain: Simple Example

- Probability of reaching E from $s_0$ is $0.2 \times 0.5 + 0.8 \times 0.1 \times 0.5 = 0.14$
- The chain has infinite paths if state graph has loops
  - Need to solve a system of linear equations to compute probabilities
PRI SM

[Kwiatkowska et al., U. of Birmingham]

◆ Probabilistic model checker

◆ System specified as a Markov chain
  - Parties are finite-state machines w/ local variables
  - State transitions are associated with probabilities
    - Can also have nondeterminism (Markov decision processes)
  - All parameters must be finite

◆ Correctness condition specified as PCTL formula

◆ Computes probabilities for each reachable state
  - Enumerates reachable states
  - Solves system of linear equations to find probabilities
module Simple
    state: [1..5] init 1;
    [] state=1 -> 0.8: state'=2 + 0.2: state'=3;
    [] state=2 -> 0.1: state'=3 + 0.9: state'=4;
    [] state=3 -> 0.5: state'=4 + 0.5: state'=5;
endmodule

IF state=3 THEN with prob. 50% assign 4 to state, with prob. 50% assign 5 to state
Modeling Crowds with PRISM

- Model probabilistic path construction
- Each state of the model corresponds to a particular stage of path construction
  - 1 router chosen, 2 routers chosen, ...
- Three probabilistic transitions
  - Honest router chooses next router with probability $p_f$, terminates the path with probability $1-p_f$
  - Next router is probabilistically chosen from N candidates
  - Chosen router is hostile with certain probability
- Run path construction protocol several times and look at accumulated observations of the intruder
module crowds

// N = total # of routers, C = # of corrupt routers
// badC = C/N, goodC = 1-badC
[!] (!good & !bad & run) ->
  goodC: (good’=true) & (revealAppSender’=true) &
          (run’=false) +
  badC: (badObserve’=true) & (run’=false);

// Forward with probability PF, else deliver
[!] (good & !deliver) ->
  PF: (pIndex’=pIndex+1) & (forward’=true) &
      (good’=false) +
  notPF: (deliver’=true);

endmodule
module crowds

. . .

// Record the apparent sender and deliver
[] (badObserve & appSender=0) ->
   (observe0'=observe0+1) & (deliver'=true);

. . .

// Record the apparent sender and deliver
[] (badObserve & appSender=15) ->
   (observe15'=observe15+1) & (deliver'=true);

. . .
endmodule

- For each observed path, bad routers record apparent sender
- Bad routers collaborate, so treat them as a single attacker
- No cryptography, only probabilistic inference
PCTL Logic

- Probabilistic Computation Tree Logic
- Used for reasoning about probabilistic temporal properties of probabilistic finite state spaces
- Can express properties of the form “under any scheduling of processes, the probability that event E occurs is at least p”
  - By contrast, Mur\(\phi\) can express only properties of the form “does event E ever occur?”
PCTL Syntax

◆ State formulas
  • First-order propositions over a single state

\[ \Phi ::= \text{True} \mid a \mid \Phi \land \Phi \mid \Phi \lor \Phi \mid \neg \Phi \mid P_{\geq p}[\Psi] \]

◆ Path formulas
  • Properties of chains of states

\[ \Psi ::= X \Phi \mid \Phi U^{\leq k} \Phi \mid \Phi U \Phi \]

- Predicate over state variables (just like a Murϕ invariant)
- Path formula holds with probability > p
- State formula holds for next state in the chain
- First state formula holds for every state in the chain until second becomes true
A state formula is a first-order state predicate

- Just like non-probabilistic logic

\[
\phi = (y > 1) \, | \, (x = 1)
\]
A path formula is a temporal property of a chain of states

Φ₁ U Φ₂ = "Φ₁ is true until Φ₂ becomes and stays true"

ψ = (y>0) U (x>y) holds for this chain
PCTL: Probabilistic State Formulas

- Specify that a certain predicate or path formula holds with probability no less than some bound

\[ \phi = P_{>0.5}[ (y>0) \cup (x=2) ] \]
module crowds

    // Record the apparent sender and deliver
    [] (badObserve & appSender=0) ->
        (observe0'=observe0+1) & (deliver'=true);

    // Record the apparent sender and deliver
    [] (badObserve & appSender=15) ->
        (observe15'=observe15+1) & (deliver'=true);

endmodule

Every time a hostile crowd member receives a message from some honest member, he records his observation (increases the count for that honest member)
Negation of Probable Innocence

launch $\rightarrow$

$[true \cup (observe0 > observe1) \ & \ done] > 0.5$

... 

launch $\rightarrow$

$[true \cup (observe0 > observe9) \ & \ done] > 0.5$

“The probability of reaching a state in which hostile crowd members completed their observations and observed the true sender (crowd member #0) more often than any of the other crowd members (#1 … #9) is greater than 0.5”
Analyzing Multiple Paths with PRI SM

Use PRI SM to automatically compute interesting probabilities for chosen finite configurations

◆ “Positive”: $P(K_0 > 1)$
  • Observing the true sender more than once

◆ “False positive”: $P(K_{i \neq 0} > 1)$
  • Observing a wrong crowd member more than once

◆ “Confidence”: $P(K_{i \neq 0} \leq 1 | K_0 > 1)$
  • Observing only the true sender more than once

$K_i = \text{how many times crowd member } i \text{ was recorded as apparent sender}$
Size of State Space

All hostile routers are treated as a single router, selected with probability 1/6
Sender Detection (Multiple Paths)

- All configurations satisfy probable innocence
- Probability of observing the true sender increases with the number of paths observed...
- ...but decreases with the increase in crowd size
- Is this an attack?
  - Can’t avoid building new paths
  - Hard to prevent attacker from correlating same-sender paths

1/6 of routers are hostile
Attacker’s Confidence

- “Confidence” = probability of detecting only the true sender
- Confidence grows with crowd size
- Maybe this is not so strange
  - True sender appears in every path, others only with small probability
  - Once attacker sees somebody twice, he knows it’s the true sender
- Is this an attack?
  - Large crowds: lower probability to detect senders, but higher confidence that the detected user is the true sender

1/6 of routers are hostile