Monitoring the full range of ω-regular properties of Stochastic Systems

Kalpana Gondi, Yogesh K. Patel, A. Prasad Sistla

University of Illinois at Chicago

Outline of the talk

- Motivation

Outline of the talk

- Motivation
- Monitoring Stochastic Systems

Outline of the talk

- Motivation
- Monitoring Stochastic Systems
- Deterministic, Probabilistic, Hybrid Algorithms

Outline of the talk

- Motivation
- Monitoring Stochastic Systems
- Deterministic, Probabilistic, Hybrid Algorithms
- Implementation

Motivation

- A component C, not thouroughly tested/verified.

Motivation

- A component C, not thouroughly tested/verified.
- C may exhibit computations that violate the correctness spec Φ

Motivation

- A component C, not thouroughly tested/verified.
- C may exhibit computations that violate the correctness spec Φ
- Need a monitor M that detects incorrect computations at run time

Motivation

- A component C, not thouroughly tested/verified.
- C may exhibit computations that violate the correctness spec Φ
- Need a monitor M that detects incorrect computations at run time
- M observes the computation of C and checks for violation of Φ

Another Motivation

- Liveness of C verified assuming fairness.

Another Motivation

- Liveness of C verified assuming fairness.

- Need to monitor C for violation of liveness or fairness.

Solution

- If Φ is Safety Property then easy ([AS85,Si85,Si87,KV99])

Solution

- If Φ is Safety Property then easy ([AS85,Si85,Si87,KV99])
- How to monitor general Φ??

Solution

- If Φ is Safety Property then easy ([AS85,Si85,Si87,KV99])
- How to monitor general Φ??
- Φ— conjunction of a safety and a liveness property

Solution

- If Φ is Safety Property then easy ([AS85,Si85,Si87,KV99])
- How to monitor general Φ??
- Φ— conjunction of a safety and a liveness property
- Over approximate Φ by a safety property [AR05] (Liberal Monitor)

Solution

- If Φ is Safety Property then easy ([AS85,Si85,Si87,KV99])
- How to monitor general Φ??
- Φ— conjunction of a safety and a liveness property
- Over approximate Φ by a safety property [AR05] (Liberal Monitor)
- Under approximate it by a safety property [MSSZ05,SZZ06] (Conservative Monitor)

Comparison to earlier work

- Earlier Work [SS08]: Deterministic Monitors for the case Φ is a Det. Buchi automaton.

Comparison to earlier work

- Earlier Work [SS08]: Deterministic Monitors for the case Φ is a Det. Buchi automaton.
- New Work:

Comparison to earlier work

- Earlier Work [SS08]: Deterministic Monitors for the case Φ is a Det. Buchi automaton.
- New Work:
  - Φ is a Det. Streett automaton— all ω-regular properties.

Comparison to earlier work

- Earlier Work [SS08]: Deterministic Monitors for the case Φ is a Det. Buchi automaton.
- New Work:
  - Φ is a Det. Streett automaton— all ω-regular properties.
  - Accurate Deterministic, Probablistic and Hybrid Algs.

Comparison to earlier work

- Earlier Work [SS08]: Deterministic Monitors for the case Φ is a Det. Buchi automaton.
- New Work:
  - Φ is a Det. Streett automaton— all ω-regular properties.
  - Accurate Deterministic, Probablistic and Hybrid Algs.
  - Implementation: Tool– Stochastic Monitor(SM)

Monitoring Stochastic Systems

A *Hidden Markov Chain (HMC)* is a pair (G, O) where

- $G = (S, R, \phi)$ is a finite Markov chain;

Monitoring Stochastic Systems

A *Hidden Markov Chain (HMC)* is a pair (G, O) where

- $G = (S, R, \phi)$ is a finite Markov chain;
- $O : S \rightarrow \Sigma$ is an output function

Monitoring Stochastic Systems

A *Hidden Markov Chain (HMC)* is a pair (G, O) where

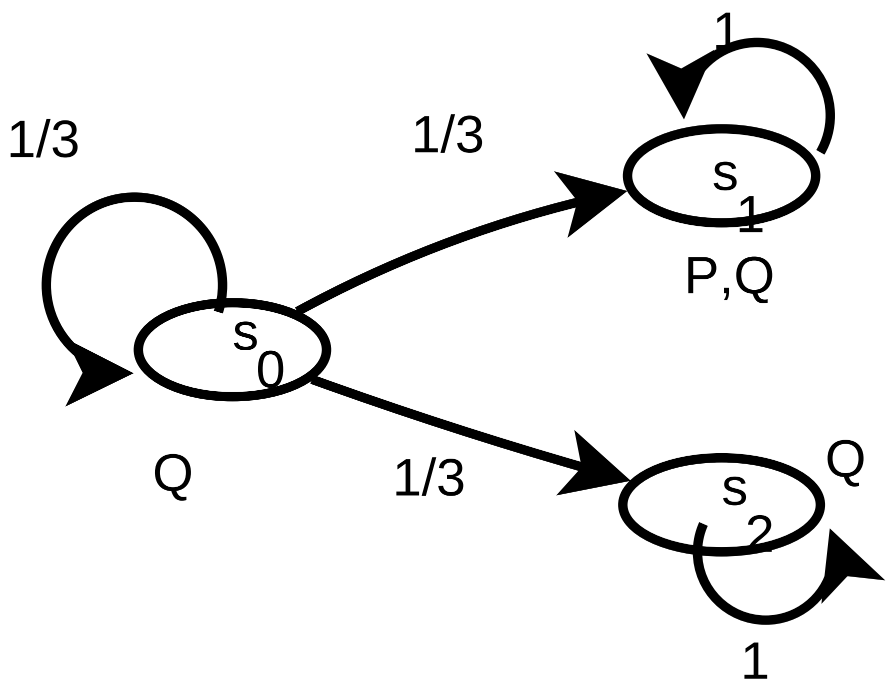
- $G = (S, R, \phi)$ is a finite Markov chain;
- $O : S \rightarrow \Sigma$ is an output function
- $\Sigma = 2^{\mathcal{P}}$, $\mathcal{P}$– set of atomic propositions

Monitoring Stochastic Systems

A *Hidden Markov Chain (HMC)* is a pair (G, O) where

- $G = (S, R, \phi)$ is a finite Markov chain;
- $O : S \rightarrow \Sigma$ is an output function
- $\Sigma = 2^{\mathcal{P}}$, $\mathcal{P}$– set of atomic propositions
- Define $\mathcal{E}$— the class of measurable subsets of Σ^ω— as the smallest set so that
  - For every $\alpha \in \Sigma^*$, $\alpha\Sigma^\omega \in \mathcal{E}$.
  - Closed under complementation and countable union.

Example



For any state s, $\mathcal{F}_s$ defines a probability measure on $\mathcal{E}$.
$\mathcal{F}_{s_0}(\Diamond P) = \frac{1}{2}$.

Accuracy of a Monitor

- The system is given by a HMC $\mathcal{H}$ which is known.

Accuracy of a Monitor

- The system is given by a HMC $\mathcal{H}$ which is known.
- Outputs of $\mathcal{H}$ are observable but not the state

Accuracy of a Monitor

- The system is given by a HMC $\mathcal{H}$ which is known.
- Outputs of $\mathcal{H}$ are observable but not the state
- Correctness spec given by a det. Streett automaton $\mathcal{A}$

Accuracy of a Monitor

- The system is given by a HMC $\mathcal{H}$ which is known.
- Outputs of $\mathcal{H}$ are observable but not the state
- Correctness spec given by a det. Streett automaton $\mathcal{A}$
- Acceptance condition of $\mathcal{A}$: Pairs of subsets $(RED, GREEN)$

Accuracy of a Monitor

- The system is given by a HMC $\mathcal{H}$ which is known.
- Outputs of $\mathcal{H}$ are observable but not the state
- Correctness spec given by a det. Streett automaton $\mathcal{A}$
- Acceptance condition of $\mathcal{A}$: Pairs of subsets $(RED, GREEN)$
- Construct a monitor $\mathcal{M}$ so that
  - $L(\mathcal{M}) \subseteq L(\mathcal{A})$.
  - $L(\mathcal{M})$ is a safety property.

Accuracy of a Monitor

- The system is given by a HMC $\mathcal{H}$ which is known.
- Outputs of $\mathcal{H}$ are observable but not the state
- Correctness spec given by a det. Streett automaton $\mathcal{A}$
- Acceptance condition of $\mathcal{A}$: Pairs of subsets $(RED, GREEN)$
- Construct a monitor $\mathcal{M}$ so that
  - $L(\mathcal{M}) \subseteq L(\mathcal{A})$.
  - $L(\mathcal{M})$ is a safety property.
- (Acceptance) Accuracy of $\mathcal{M}$ is the conditional probability $\mathcal{F}_{s_0}(L(\mathcal{M}) \mid L(\mathcal{A}))$— s_0 initial system state.

Monitoring Algorithms

- Preprocessing

Monitoring Algorithms

- Preprocessing
  - Compute Markov chain G'— the product of G and $\mathcal{A}$.

Monitoring Algorithms

- Preprocessing
  - Compute Markov chain G'— the product of G and $\mathcal{A}$.
  - A state (s, q) in G' is good if $\mathcal{F}_s(L(\mathcal{A}_q)) = 1$ and bad if $\mathcal{F}_s(L(\mathcal{A}_q))$ is 0. $\mathcal{A}_q$ same as $\mathcal{A}$ with starting state q.

Monitoring Algorithms

- Preprocessing
  - Compute Markov chain G'— the product of G and $\mathcal{A}$.
  - A state (s, q) in G' is good if $\mathcal{F}_s(L(\mathcal{A}_q)) = 1$ and bad if $\mathcal{F}_s(L(\mathcal{A}_q))$ is 0. $\mathcal{A}_q$ same as $\mathcal{A}$ with starting state q.
  - Compute good and bad states of G'.

Monitoring Algorithms

- Preprocessing
  - Compute Markov chain G'— the product of G and $\mathcal{A}$.
  - A state (s, q) in G' is good if $\mathcal{F}_s(L(\mathcal{A}_q)) = 1$ and bad if $\mathcal{F}_s(L(\mathcal{A}_q))$ is 0. $\mathcal{A}_q$ same as $\mathcal{A}$ with starting state q.
  - Compute good and bad states of G'.
- Simulates $\mathcal{A}$ on the sequence of system outputs .

Deterministic Monitoring Algorithm

- Maintains the following variables

Deterministic Monitoring Algorithm

- Maintains the following variables
  - X: possible system states, initialized to $\{s_0\}$.

Deterministic Monitoring Algorithm

- Maintains the following variables
  - X: possible system states, initialized to $\{s_0\}$.
  - q: the automaton state, initialized to q_0.

Deterministic Monitoring Algorithm

- Maintains the following variables
  - X: possible system states, initialized to $\{s_0\}$.
  - q: the automaton state, initialized to q_0.
  - i: denotes the number of times an accepting automaton state is reached. Initialized to 0.

Deterministic Monitoring Algorithm

- Maintains the following variables
  - X: possible system states, initialized to $\{s_0\}$.
  - q: the automaton state, initialized to q_0.
  - i: denotes the number of times an accepting automaton state is reached. Initialized to 0.
  - $counter$: denotes the number of expected outputs before an accepting automaton state.

Det. Alg. Continued

- Procedure $GetInputAndUpdate()$:
  - Get next input from the system;
  - Simulate $\mathcal{A}$ for one step and Update q as well as X;
  - **If** all states in $X \times \{q\}$ are good **then** accept;
  - **If** all states in $X \times \{q\}$ are bad **then** reject;

Deterministic Algorithm Contd

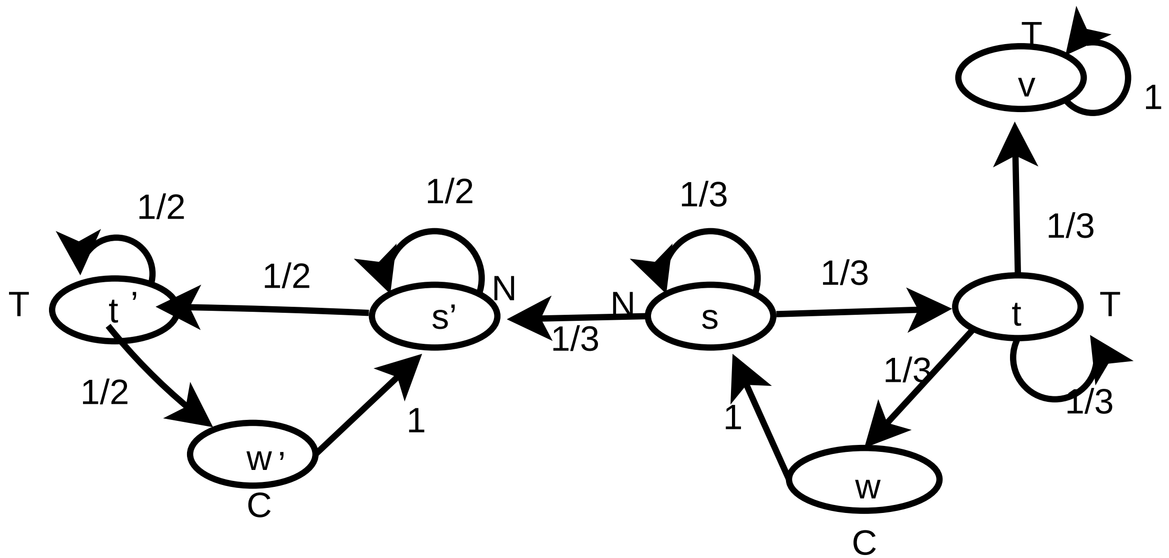
Loop forever

- GetInputAndUpdate();
- **If** $q \in RED$ **then** $counter := counter - 1$;
- **If** $counter = 0$ **then** reject;
- **If** $q \in GREEN$ **then** $\{i := i + 1;\ counter := f(q, X, i)\}$

- **Theorem:** For any y, $0 \leq y < 1$, there exists a constant c such that if $f(q, X, i) = c \cdot i$ then the acceptance accuracy of the monitor is at least y.

- **Theorem:** If the HMC is fully visible, then the monitor can be simplified to have acceptance accuracy to be 1.

Example:Resource Acquisition



- s is the initial state.
- v— the state where the server crashed.
- Property to be monitored— $\Box(T \to \Diamond C)$.
- Acceptance accuracy of 0.9 can be achieved by choosing $k = 3$.

Probabilistic Algorithm

- *Uses* probability variable p instead of $counter$.
- p initialized to probability value p_0.
- Uses variables X, q as before.

Loop forever

- GetInputAndUpdate();
- **If** $q \in RED$ **then** reject with probability p;
- **If** $q \in GREEN$ **then** $p := \frac{p}{c}$

Probabilistic Algorithm Contd.

- **Theorem:** For any y, $0 \leq y < 1$, there exists constants p_0, c for which the acceptance accuracy of the monitor is at least y.

Hybrid Algorithm

- Combines both deterministic, probabilistic algs.
- Uses variable $counter$ initialized to k.
- Uses variables X, q as before.

Hybrid Algorithm Contd.

Loop forever

- GetInputAndUpdate();
- **If** $q \in RED$ **then** $counter--$, Toss a fair coin;
- **If** $counter = 0$ **then**

  If last k coin tosses were tails **then** reject
  Else $counter := k$;

- **If** $q \in GREEN$ **then** $counter := ++k$

Hybrid Algorithm Contd.

- **Theorem:** For any y, $0 \leq y < 1$, there exists an initail counter value such that the acceptance accuracy of the monitor is at least y.

Implementation

- Developed a tool : SM (Stochastic Monitor)
- Input: high level description of a synch. probabilistic program;
- Uses PRISM tool to obtain the Markov chain $\mathcal{M}$;
- Takes automaton $\mathcal{A}$ as another input;
- Constructs product Markov Chain $\mathcal{M}'$;
- Computes its good, bad product states;
- Generates a monitor using other parameters.

Experimental Results

- Considered three examples;
- Peterson's Mutual Excl Alg: Second process can die in the critical section;
- Property Monitored: $\Box(T_1 \rightarrow \Diamond C_1)$;

- Mutual Excl with Semaphores: Second can die in the critical section
- Property Monitored: $\Box\Diamond T_1 \supset \Box\Diamond C_1$;

- Bounded Retransmission Protocol: Packets can be lost in transmission;
- Property Monitored: The file will eventually be transmitted.

Experimental results Contd

Deterministic			Probabilistic			Hybrid		
Counter	Accuracy	Time(mS)	Counter	Accuracy	Time	Counter	Accuracy	Time
4	16	130.27	0.25	23	151.3	4	80	1580.21
8	33	386.43	0.13	30	408.69	8	96	2816.9
16	50	770.02	0.06	50	827.06	16	100	3179.77
32	53	771.65	0.03	63	1796.08	24	100	2909.28

Table 1: Peterson's Mutual Exclusion Algorithm

Deterministic			Probabilistic			Hybrid		
Counter	Accuracy	Time(mS)	Counter	Accuracy	Time	Counter	Accuracy	Time
4	37	725.6	0.25	11	242.2	4	83	1855.5
8	62	1215.8	0.13	35	830.85	8	92	2221.3
16	70	1359.4	0.06	57	1375.8	16	100	2399
32	83	1596	0.03	80	1974.4	24	100	2526

Table 2: Mutual Exclusion Algorithm using Semaphores

Experimental Results Contd

Deterministic			Probabilistic			Hybrid		
Counter	Accuracy	Time(mS)	Counter	Accuracy	Time	Counter	Accuracy	Time
12	10	0.31	0.13	16	0.14	3	50	0.39
16	63	0.45	0.06	40	0.34	4	80	0.40
20	93	0.47	0.03	66	0.44	6	96	0.51
24	96	0.51	0.02	76	0.44	8	100	0.62

Table 3: Bounded Retransmission Protocol (with N=2 & Max=4)

Related Work

- Our [SS08] paper gave det algs for det .Buchi automata
- Monitoring for safety properties done by many people [Si87], [KV99], etc.
- Recent work— Amorium and Rosu (CAV2005)– handle some liveness. Concentrate on evaluating efficiently atomic propositions in system states.
- The paper [PZZ 2005] uses game theoretic approach.

Conclusion

- Need to extend to Hidden Markov Decision Processes to handle asynchronous concurrency
- Other cost measures for tuning deterministic algs for HMCs.
- How to monitor for complex systems? Use Assume/guarantee paradigms.