Contributions to Statistical Model Checking

Axel Legay

18 novembre 2015

Jury: Radu Grosu, RAPPORTEUR
      Tiziana Margaria, RAPPORTEUR
      Sylvain Peyronnet, RAPPORTEUR
      Albert Benveniste, EXAMINATEUR
      Kim. G. Larsen, EXAMINATEUR
      Philippe Baufreton, EXAMINATEUR
      Jean-Marc Jezequel, PRÉSIDENT DU JURY
Computer Systems

- Present everywhere: phone, car, plane, medical device, ...

- Errors have physical and/or economy impacts

- **Major quality** indications: absence of errors

- **Problem:** more and more complex, hence **difficult to validate**

- **Consequence:** Industry devotes large amount of time and money to validation.
Validation Techniques

Examples:
- Code reviewing: rates are about 150 lines of code per hour. Inspecting and reviewing more than a few hundred lines of code per hour for critical software (such as safety critical embedded software) may be too fast to find errors.
- Testing: massively used, but faces the coverability problem and has problems with non determinism and quantities.
- ...

Another approach **Model Checking (Clarke, Emerson, Sifakis):**
- Automated and exhaustive technique
- Relies on mathematical models for requirements and systems
- May suffer from state-space explosion.
Model Checking: overview

- Requirements
- Formalizing
- Property Specification
- Temporal Logic
- Satisfied
- Model Checker
- System
- Modeling
- System model
- Violation + counterexample
- (Probabilistic) system

Axel Legay (Inria) Contributions to Statistical Model Checking 18th November 2015
Model Checking for Stochastic Systems

Systems that contain probabilistic features. **Central in:**

- Hardware modeling
- Systems biology
- Social and economical systems
- ...

**Question:** What is the probability to satisfy a requirement?

**Problem:** State-space explosion

**Solution:** Statistical Model Checking (between test and model checking)
Model Checking for Stochastic Systems

Systems that contain probabilistic features. Central in:

- Hardware modeling
- Systems biology
- Social and economical systems
- ...

**Question:** What is the probability to satisfy a requirement?

**Problem:** State-space explosion

**Solution:** Statistical Model Checking (between test and model checking)
Statistical Model Checking (Youness, Larsen, Peyronnet)

Parameters:
- Number of samples,
- Confidence,
- Precision

S Simulator executable

Φ Monitor executable

Produces an execution trace

SMC core executable

OK/KO local verdict

collectstriggers

Final verdict
Statistical Model Checking (Youness, Larsen, Peyronnet)

- Easily parallelisable
- System S
  - S Simulator executable
  - Parameters: Number of samples, Confidence, Precision
- Property $\Phi$
  - $\Phi$ Monitor executable
  - Produces an execution trace
- SMC core
  - SSP, SPRT, PESTIM...
  - executable
  - OK/KO local verdict
  - collects
  - triggers
- Final verdict
  - Easily parallelisable
Statistical Model Checking (Youness, Larsen, Peyronnet)

- Easily parallelisable
- System S
  - S Simulator executable
  - Produces an execution trace
- Property Φ
  - Φ Monitor executable
  - OK/KO local verdict
- Parameters:
  - Number of samples,
  - Confidence,
  - Precision
- SMC core
  - SSP, SPRT, PESTIM...
  - executable
- Final verdict
- Less memory intensive than Model Checking
Statistical Model Checking (Youness, Larsen, Peyronnet)

- System $S$
- Parameter $\Phi$
- S Simulator \textit{executable}
- $\Phi$ Monitor \textit{executable}
- SMC core \textit{executable}: SSP, SPRT, PESTIM...
- Final verdict
- OK/KO local verdict
- Produces an execution trace
- Parameters: Number of samples, Confidence, Precision
- Approximately results given with confidence bounds
- Less memory intensive than Model Checking
- Easily parallelisable

Contribution to Statistical Model Checking: Axel Legay (Inria)
Requirement’s model in SMC

**BLTL (Pnueli’77):** \( \phi := \alpha | \phi \lor \phi | \phi \land \phi | \neg \phi | \Diamond t \phi | \Box t \phi | \phi U t \phi \)

- \( \Diamond \): “next” operator
- \( \Box \): “eventually” operator
- \( \Diamond \leq t \): “always” operator
- \( U \): “until” operator
- \( \Diamond \leq t(x \geq 0) \): \( x \) will be eventually greater than 0 within \( t \) time units

**Probabilistic BLTL:**

- \( P_{\sim \theta} \phi \) with \( \sim \in \{<, >, =\} \) and \( \theta \in [0, 1] \)
- \( P_{\leq 0.2}(\Diamond \leq t \Box \leq s(x \geq 0)) \).
Requirement’s model in SMC

- **BLTL (Pnueli’77):** $\phi := \alpha \mid \phi \lor \phi \mid \phi \land \phi \mid \neg \phi \mid \lozenge^t \phi \mid \Box^t \phi \mid \phi U \leq^t \phi$
  
  - $\lozenge$: “next” operator
  - $\Diamond$: “eventually” operator
  - $\Box$: “always” operator
  - $U$: “until” operator
  - $\Diamond \leq^t (x \geq 0)$: $x$ will be eventually greater than 0 within $t$ time units

- **Probabilistic BLTL:**
  
  - $P_{\sim} \phi$ with $\sim \in \{<, >, =\}$ and $\theta \in [0, 1]$
  - $P_{\leq 0.2}(\Diamond \leq^t \Box \leq^s (x \geq 0))$. 
Quantitative Algorithm: Monte Carlo estimation

- $\Omega$ a set of paths ending uniformly in a square
- $z(\omega) = 1$ if $\omega$ ends in $A$, 0 otherwise.
- $\gamma = P(\text{"end in } A\text{"}).$
- Monte-Carlo estimator:

$$\tilde{\gamma}_N = \frac{1}{N} \sum_{i=1}^{N} z(\omega_i)$$

- $$P(|\tilde{\gamma}_N - p| \geq \epsilon) \leq \alpha$$
- $$N = \lceil (\ln 2 - \ln \alpha)/(2\epsilon^2) \rceil$$
Quantitative Algorithm: Monte Carlo estimation

- $\Omega$ a set of paths ending uniformly in a square
- $z(\omega) = 1$ if $\omega$ ends in $A$, 0 otherwise.
- $\gamma = P(\text{"end in } A\text{"})$.
- Monte-Carlo estimator:

$$\tilde{\gamma}_N = \frac{1}{N} \sum_{i=1}^{N} z(\omega_i)$$

- $P(|\tilde{\gamma}_N - p| \geq \epsilon) \leq \alpha$

$$N = \lceil (\ln 2 - \ln \alpha)/(2\epsilon^2) \rceil$$
Quantitative Algorithm: Monte Carlo estimation

- $\Omega$ a set of paths ending uniformly in a square
- $z(\omega) = 1$ if $\omega$ ends in $A$, $0$ otherwise.
- $\gamma = P(\text{"end in } A\text{"})$.
- Monte-Carlo estimator:

$$\tilde{\gamma}_N = \frac{1}{N} \sum_{i=1}^{N} z(\omega_i)$$

- $\Pr(|\tilde{\gamma}_N - p| \geq \varepsilon) \leq \alpha$

- $N = \lceil (\ln 2 - \ln \alpha)/(2\varepsilon^2) \rceil$
Qualitative Algorithm: Hypothesis Testing (Wald’45)

\( H_0 : p \leq \theta, H_1 : p > \theta \)

Compute

\[
W = \prod_{i=1}^{m} \frac{Pr(Z_i = z_i \mid p = \theta - \delta)}{Pr(Z_i = z_i \mid p = \theta + \delta)} = \frac{(\theta - \delta)^{d_m} (1 - \theta + \delta)^{m-d_m}}{(\theta + \delta)^{d_m} (1 - \theta - \delta)^{m-d_m}}, \tag{1}
\]

where \( d_m = \sum_{i=1}^{m} z_i \).

Stop when :

- \( W \geq (1 - \beta)/\alpha \): \( H_1 \) is accepted;
- \( W \leq \beta/(1 - \alpha) \): \( H_0 \) is accepted, with
- \( \alpha \) (resp. \( \beta \)) is the probability to accept \( H_1 \) (resp. \( H_0 \)) when \( H_0 \) (resp. \( H_1 \) holds).
Qualitative Algorithm: Hypothesis Testing (Wald’45)

\[ H_0 : p \leq \theta, \ H_1 : p > \theta \]

Compute

\[
W = \prod_{i=1}^{m} \frac{Pr(Z_i = z_i \mid p = \theta - \delta)}{Pr(Z_i = z_i \mid p = \theta + \delta)} = \frac{(\theta - \delta)^{d_m} (1 - \theta + \delta)^{m - d_m}}{(\theta + \delta)^{d_m} (1 - \theta - \delta)^{m - d_m}},
\]

where \( d_m = \sum_{i=1}^{m} z_i \).

Stop when :

- \( W \geq (1 - \beta)/\alpha \) : \( H_1 \) is accepted;
- \( W \leq \beta/(1 - \alpha) \) : \( H_0 \) is accepted, with
- \( \alpha \) (resp. \( \beta \)) is the probability to accept \( H_1 \) (resp. \( H_0 \)) when \( H_0 \) (resp. \( H_1 \) holds).
Contributions

- **Plasma**: A new modular tool chain:
  - An effort to unify SMC initiatives

- **Quantitative Algorithms**: to estimate the probability of an event
  - Rare Event Simulation, towards bug finding

- **Qualitative Algorithms**: to compare the probability with a given threshold
  - Change detection Algorithms

- **Extension to more complex models**:
  - Timed Systems (collaboration with aau over 6 years)
  - Systems of Systems
  - Markov Decision Processes
Contributions

- **Plasma: A new modular tool chain:**
  - An effort to unify SMC initiatives

- **Quantitative Algorithms:** to estimate the probability of an event
  - Rare Event Simulation, towards bug finding

- **Qualitative Algorithms:** to compare the probability with a given threshold
  - Change detection Algorithms

- **Extension to more complex models:**
  - Timed Systems (collaboration with aau over 6 years)
  - Systems of Systems
  - Markov Decision Processes
Contributions

- **Plasma: A new modular tool chain:**
  - An effort to unify SMC initiatives

- **Quantitative Algorithms:** to estimate the probability of an event
  - Rare Event Simulation, towards bug finding

- **Qualitative Algorithms:** to compare the probability with a given threshold
  - Change detection Algorithms

- **Extension to more complex models:**
  - Timed Systems (collaboration with aau over 6 years)
  - Systems of Systems
  - Markov Decision Processes
Contributions

- **Plasma: A new modular tool chain:**
  - An effort to unify SMC initiatives

- **Quantitative Algorithms:** to estimate the probability of an event
  - Rare Event Simulation, towards bug finding

- **Qualitative Algorithms:** to compare the probability with a given threshold
  - Change detection Algorithms

- **Extension to more complex models:**
  - Timed Systems (collaboration with aau over 6 years)
  - Systems of Systems
  - Markov Decision Processes
Contributions

- **Plasma: A new modular tool chain:**
  - An effort to unify SMC initiatives

- **Quantitative Algorithms:** to estimate the probability of an event
  - Rare Event Simulation, towards bug finding

- **Qualitative Algorithms:** to compare the probability with a given threshold
  - Change detection Algorithms

- **Extension to more complex models:**
  - Timed Systems (collaboration with aau over 6 years)
  - Systems of Systems
  - Markov Decision Processes
Outline

1. Introduction

2. Plasma: a social application
   - Architecture of the tool
     - Application
     - Future Work

3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work

4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work

5. Conclusion
Plasma Lab: Generic Statistical Model Checker

- New SMC algorithms: MC, HT, Isplit, Ispl, CUSUM . . .
- Plugins: RML, SystemC, Simulink, Biological, BLTL, GCSL . . .
- Distributed Algorithms
- Applications: social sciences, SoS, systems biology, ...
- 5 EU and 3 national projects + collaboration with industry partners

Architecture
Outline

1 Introduction

2 Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work

3 Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work

4 Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work

5 Conclusion
The Dali project: A concrete PLASMA application

Objective: to design C-Walker, a trolley robot used to drive an old lady from a source to a destination in a commercial center.

Constraints:
- Environment populated by known and unknown fixed obstacles, pedestrians and other moving objects
- Must respect user preferences / constraints (distance with shops, humans, ...).

Challenge:
- Must minimise probability of accidents and stress to user.

Contribution: The design of the planning algorithm.
Two Planning Algorithms

- Long term planning via Dijkstra algorithm
- Short term planning via SMC (this thesis)
Short term planner

Receives:
- Global plan and preferences
- Sensors:
  - locate user, other pedestrians and local fixed objects
  - estimate velocity of other pedestrians

Challenge:
- Suggests user’s next move.

Solution: Apply SMC
- Q1: How to represent the environment/trolley?
- Q2: How to represent the objectives?
Solutions to Question 1: Social Forces Model (SFM) for the environment (Helbing’2000)

Models human desires as forces

- Includes notions of objectives, attractions, repulsions and unpredictable behaviour
- Desired velocity $v_i^0$ to follow global path
- **Suggestion**: Modify the desired velocity to propose adequate move
- Here is the model for individual $i$:

$$\frac{dv_i}{dt} = \frac{v_i^0 - v_i}{\tau_i} + \frac{f_i + \xi_i}{m_i}$$

$$\frac{dx_i}{dt} = v_i$$

$v_i^0$ – desired velocity
$v_i$ – actual velocity
$f_i$ – social and physical forces
$\xi_i$ – random movement
$x_i$ – actual location
$\tau_i$ – reaction time
$m_i$ – mass
Solution to Question 2: Global objectives via BLTL

Principle:

- BLTL encodes high level objectives
  - minimum distance, maximum time, etc.
  \[
  (G_{[0,4]} \land i \neq u |x_u - x_i| > 0.5) \land (F_{[0,4]} |x_u - w| < 0.2)
  \]
- BLTL properties verified on simulations provided by SFM
Local planning algorithm via SMC

- Each individual $i$ and the lady are represented by their social force models.
- The algorithm modifies the desired velocity $v^0$ of the lady by modifying its value angle (direction).

For each value $\epsilon \in \{-60, -30, -15, 0, +15, +30, +60\}$, do:
  - Compute a virtual waypoint that is displaced from the real: $\epsilon' = \epsilon \frac{t_{\text{max}} - t}{t_{\text{max}}}$
  - Specify $v^0$ by direct path to the virtual waypoint at users average speed.
  - Simulate multiple paths of length $t_{\text{max}}$ time to apply Monte Carlo.

**Outcome:** The smallest angular deviation that maximises probability of success, i.e., the best direction.

**Observation:** SMC may correct the social force model.
Model of local environment

- Position of user located by sensors
- Position and velocity of other pedestrians detected by sensors
- Local waypoint on global plan
SFM predicts collision

social force model includes stochasticity to model unpredictable behaviour

short term planner tests many plausible futures predicted by the social force model, using SMC

short term planner estimates probability that future path will respect user's minimum proximity
Cognitive engine suggests detour

short term planner tests many immediate course changes to estimate probability of collision

suggests minimum course change for acceptable probability of collision

gfailure to find acceptable local path triggers long term planner
Performances

Implemented on embedded device (Beagleboard xM)

- Ability to re-plan every 500ms (with no optimisations)
  - 50 simulation for $\epsilon = 0.2$ and $\alpha = 0.9$
  - $\approx 92\%$ of the time for simple scenario
  - $\approx 55\%$ of the time for complex scenario
Obstacle avoidance scenario (3 agents, 1 obstacle)

<table>
<thead>
<tr>
<th>$t_{\text{max}}$ (s)</th>
<th><strong>SMC + SFM</strong></th>
<th><strong>SMC + LIN</strong></th>
<th>SFM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>$P_{\text{safe}}$</td>
<td>0.74</td>
<td>0.89</td>
<td>0.99</td>
</tr>
<tr>
<td>$\Delta$ (m)</td>
<td>0.35</td>
<td>0.99</td>
<td>1.44</td>
</tr>
</tbody>
</table>

Shopping scenario (7 agents, 6 counters)

<table>
<thead>
<tr>
<th>$t_{\text{max}}$ (s)</th>
<th><strong>SMC + SFM</strong></th>
<th><strong>SMC + LIN</strong></th>
<th>SFM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>$P_{\text{safe}}$</td>
<td>0.99</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$\Delta$ (m)</td>
<td>0.77</td>
<td>0.79</td>
<td>0.65</td>
</tr>
</tbody>
</table>

$SMC$ – statistical model checking  
$SFM$ – social force model  
$LIN$ – linear interpolation model  
$P_{\text{safe}}$ – probability of not colliding  
$\Delta$ – average deviation from global plan
Outline

1. Introduction

2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work

3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work

4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work

5. Conclusion
Future Work

Improve the efficiency:
- Distributed SMC (GPGPU).

ACANTO project:
- group motion planning
- motion planning for social activities
- motion planning to explicitly benefit health

Highly visible project: national press, euronews, ...
Outline

1. Introduction
2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work
3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work
4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work
5. Conclusion

Axel Legay (Inria)
Contributions to Statistical Model Checking
18th November 2015
The Rare Event problem in SMC

Rare events (computing very small probabilities) are challenging

- Require a lot of samples (to see the event at least once)
- Relative error explodes

How to overcome the Rare Event problem in SMC?

- **Importance Sampling**: Tackle the problem by reasoning on the model (ex. Ridder)
- **Importance Splitting**: Tackle the problem by reasoning on the property (ex. Cerou-Guyader)

We worked on both, and this habilitation focuses on the second one.
Basics of Importance Splitting

Let $A$ be a rare event and $(A_k)_{0 \leq k \leq n}$ be a sequence of nested events:

$$A_0 \supset A_1 \supset \ldots \supset A_n = A$$

By Bayes formula,

$$\gamma \overset{\text{def}}{=} P(A) = P(A_0)P(A_1 \mid A_0)P(A_2 \mid A_1)\ldots P(A_n \mid A_{n-1})$$

With:

$$\forall k, \ P(A_k \mid A_{k-1}) = \gamma_k \geq \gamma$$

In Model Checking: Each $A_i$ is e.g. a BLTL property $\varphi_i$, such that

$$\varphi = \varphi_n \Rightarrow \varphi_{n-1} \Rightarrow \cdots \Rightarrow \varphi_0$$
Outline

1. Introduction

2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work

3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work

4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work

5. Conclusion
Importance Splitting estimator $\gamma$ for $\varphi$

- Write $\gamma$ as a product of $\gamma_k$ (for $\varphi_k$)
- Estimate separately each $\gamma_k$

Questions:

- How to split a property in $\varphi_k$ (existential and universal properties)?
- How to estimate $\gamma_k$ for each $\varphi_k$ (levels)?
- How is the confidence interval?
Example: Reaching Level 3 within $T$ (Guyader for MC)

- One level per $\varphi_k$
- $\gamma_k$ is the number of simulations that satisfy $\varphi_k$ divided by the total number.
Example: Reaching Level 3 within $T$ (Guyader for MC)

- One level per $\varphi_k$
- $\gamma_k$ is the number of simulations that satisfy $\varphi_k$ divided by the total number.
Example: Reaching Level 3 within T (Guyader for MC)

- One level per $\varphi_k$
- $\gamma_k$ is the number of simulations that satisfy $\varphi_k$ divided by the total number.
Example: Reaching Level 3 within T (Guyader for MC)

- One level per $\varphi_k$
- $\gamma_k$ is the number of simulations that satisfy $\varphi_k$ divided by the total number.

\[
P(\text{reaching Level 3}) = \frac{3}{5} \times \frac{2}{5} \times \frac{2}{5}
\]
(1 – \(\alpha\)) CI based on relative variance \(\sigma^2\):

\[
\left[ \tilde{\gamma} \left( \frac{1}{1 + \frac{z_\alpha \sigma}{\sqrt{N}}} \right); \tilde{\gamma} \left( \frac{1}{1 - \frac{z_\alpha \sigma}{\sqrt{N}}} \right) \right]
\]

with \(\sigma^2 \geq \sum_{k=1}^{n} \frac{1 - \gamma_k}{\gamma_k}\)

\(\sigma^2\) is minimize when all the \(\gamma_k\) have same probability.
Importance Splitting in a Model Checking Context and level strategy: Algorithm 1 implemented in Plasma

Initialisation:
\[ k=0, L_k = s_0 \]

Simulator
Initial state chosen randomly in \( L_k \)

Monitor \( z(\omega; k+1) \)

Initialisation:
\[ k=0, L_k = s_0 \]

ISp core

\( L_{k+1} \) distribution of prefixes satisfying \( \varphi_{k+1} \)

\[ \hat{y}_{k+1} = \frac{1}{n} \sum_{i=1}^{n} z(\omega^{(i)}; k+1) \]

\( \hat{y} = \prod_{k=1}^{m} \hat{y}_{k} \)

After \( n \) samples outputs

\( k\leftarrow k+1 \)

k\leftarrow k+1

OK/KO local verdict with respect to \( \Phi_{k+1} \)

collects

produces an execution path \( \omega \)

triggers
Example: the Dining Philosophers Problem

- property of interest:
  \( \phi = \phi_5 = F^{30} \) (Phil i eat)
  - \( \phi_4 = F^{30} \) (Phil i picks 2 forks)
  - \( \phi_3 = F^{30} \) (Phil i picks 1 fork)
  - \( \phi_2 = F^{30} \) (Phil i intends to take a fork)
  - \( \phi_1 = F^{30} \) (Phil i chooses)
  - \( \phi_0 = F^{30} \) (Phil i thinks)
  - \( \phi_5 \Rightarrow \phi_4 \Rightarrow \cdots \Rightarrow \phi_0 \)

Figure: Automata modelling a philosopher
Example: the Dining Philosophers Problem

property of interest:
\[ \phi = \phi_5 = F^{30} \text{ (Phil i eat)} \]
\[ \phi_4 = F^{30} \text{ (Phil i picks 2 forks)} \]
\[ \phi_3 = F^{30} \text{ (Phil i picks 1 fork)} \]
\[ \phi_2 = F^{30} \text{ (Phil i intends to take a fork)} \]
\[ \phi_1 = F^{30} \text{ (Phil i chooses)} \]
\[ \phi_0 = F^{30} \text{ (Phil i thinks)} \]

\[ \phi_5 \Rightarrow \phi_4 \Rightarrow \cdots \Rightarrow \phi_0 \]
Application of Algorithm 1

What we should obtain:

- 150 philosophers
- Property of interest: \( \phi = \phi_5 = F^{30} \) (Phil i eat)
- \( \gamma \approx 1.59 \times 10^{-6} \)

Results when applying Algorithm 1:

- Time (with 1000 paths per iteration): 6.95 seconds in average
- \( \tilde{\gamma}_{1:5} \in \{0.158, 0.088, 0.027, 0.008, 0.003\} \)
- \( \tilde{\gamma} = 10^{-8} \)

Conclusion: Needs to exploit a refined score level.
Application of Algorithm 1

What we should obtain:
- 150 philosophers
- property of interest: 
  \( \phi = \phi_5 = F^{30} \) (Phil i eat)
- \( \gamma \approx 1.59 \times 10^{-6} \)

Results when applying Algorithm 1:
- Time (with 1000 paths per iteration): 6.95 seconds in average
- \( \tilde{\gamma}_{1:5} \in \{0.158, 0.088, 0.027, 0.008, 0.003\} \)
- \( \tilde{\gamma} = 10^{-8} \)

Conclusion: Needs to exploit a refined score level.
Application of Algorithm 1

What we should obtain:

- 150 philosophers
- property of interest:
  \[ \phi = \phi_5 = F^{30} \text{ (Phil i eat)} \]
- \[ \gamma \approx 1.59 \times 10^{-6} \]

Results when applying Algorithm 1:

- Time (with 1000 paths per iteration): 6.95 seconds in average
- \[ \tilde{\gamma}_{1:5} \in \{0.158, 0.088, 0.027, 0.008, 0.003\} \]
- \[ \tilde{\gamma} = 10^{-8} \]

Conclusion: Needs to exploit a refined score level.
Relative variance of the estimator: \( \sigma^2 = \sum_{k=1}^{n} \frac{1 - \gamma_k}{\gamma_k} \)

For a fixed number of levels, minimal variance if all the conditional probabilities are equal (\( \gamma_0 \)).

Problems with Algorithm 1:

1. No guarantee to have nearly similar conditional probabilities (force the number of simulation to pass a given score to be equal)
2. Levels might be too coarse (replace them with adaptive scores – how fast do you satisfy the property?)
Observations

- Relative variance of the estimator: $\sigma^2 = \sum_{k=1}^{n} \frac{1-\gamma_k}{\gamma_k}$
- For a fixed number of levels, minimal variance if all the conditional probabilities are equal ($= \gamma_0$).

Problems with Algorithm 1:

1. No guarantee to have nearly similar conditional probabilities (force the number of simulation to pass a given score to be equal)
2. Levels might be too coarse (replace them with adaptive scores – how fast do you satisfy the property?)
Observations

- Relative variance of the estimator: \( \sigma^2 = \sum_{k=1}^{n} \frac{1-\gamma_k}{\gamma_k} \)

- For a fixed number of levels, minimal variance if all the conditional probabilities are equal (\( = \gamma_0 \)).

- Problems with Algorithm 1:

  1. No guarantee to have nearly similar conditional probabilities (force the number of simulation to pass a given score to be equal).
  2. Levels might be too coarse (replace them with adaptive scores – how fast do you satisfy the property?)
Solution: Adaptive IS for MC (Cerou-Guyader in Plasma)

Initialisation: \( \gamma_0, \tau_0 = 0, k=0, L_k = s_0 \)

Monitor

Initialisation:
\( \gamma_0, \tau_0 = 0, k=0, L_k = s_0 \)

Monitor

Initialisation:
\( \gamma_0, \tau_0 = 0, k=0, L_k = s_0 \)

Collects \( \{ S(\omega) \}_i \), \( S(\omega) = \max \{ S(\omega) \}_i \), \( \{ (\omega_i, t_i) \}_i \)

Axel Legay (Inria)

Contributions to Statistical Model Checking

18th November 2015 38 / 60
Outline

1. Introduction

2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work

3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work

4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work

5. Conclusion
Experimental Results given by an optimised algorithm

<table>
<thead>
<tr>
<th>Stat.</th>
<th>MC</th>
<th>Importance splitting</th>
</tr>
</thead>
<tbody>
<tr>
<td>nb exp</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>nb path</td>
<td>$10^7$</td>
<td>100</td>
</tr>
<tr>
<td>$\bar{t}$ in sec.</td>
<td>&gt; 5 h</td>
<td>1.73</td>
</tr>
<tr>
<td>$\bar{\gamma}$</td>
<td>1.5</td>
<td>1.52</td>
</tr>
<tr>
<td>$\sigma(\bar{\gamma})$</td>
<td>0.39</td>
<td>1.02</td>
</tr>
<tr>
<td>95%-CI</td>
<td>[0.74; 2.26]</td>
<td>[1.34; 1.74]</td>
</tr>
</tbody>
</table>

95%-CI based on a $3 \times 10^8$ sample: $[1.44 \times 10^{-6}; 1.72 \times 10^{-6}]$
Outline

1 Introduction

2 Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work

3 Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work

4 Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work

5 Conclusion
Future Work

- More examples
- A better comparison and a combination with importance sampling
- Applications to real-time systems
- Better comparison (or combination) with importance sampling
Outline

1. Introduction
2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work
3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work
4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work
5. Conclusion
Markov Decision Process

- **MDP** $M : (S, A, P)$
- $S$ is a set of states
- $A$ is a set of nondeterministic actions
- $P : S \times A \times S \rightarrow [0, 1]$ is probability function
  s.t. $\sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$
- Memoryless scheduler $M : S \rightarrow A$
- History-dependent scheduler for non-deterministic choices: $\mathcal{G} : S^+ \rightarrow A$
- $S^+ =$language of sequences of states $\setminus \emptyset$
- Min or Max probability for a given scheduler
Outline

1. Introduction
2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work
3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work
4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work
5. Conclusion
Solution: Sampling directly from scheduler space

- Set of actions $A$ and set of states $s \in S$ induce set of traces $S^+$
  - Implement $\mathcal{G}$ as a hash function
  - Use the function at each non-deterministic state to make a choice
  - Obtain a Markov chain that can be analyzed with SMC.

- Algorithm 1:
  - Fix a number of schedulers to explore
  - test DTMC induced by $\sigma$ with SMC (Interleaving of random choices)

- Questions:
  1. How to compute the confidence interval?
  2. What is the impact of the hash function on the scheduling algorithm?
Answer to question 1: Confidence with multiple estimates

Increase $N$ to compensate for $M$ (number of schedulers)

- Example: Chernoff bound for multiple schedulers
- Ensure all schedulers are within bound
  e.g., $N \geq \left( \ln 2 - \ln \left( 1 - \frac{M}{\sqrt{1 - \delta}} \right) \right) / (2\varepsilon^2)$
- Increase of $N$ is logarithmic in $M$
Answer to question 2: Hash functions and random numbers

Difficulty:
- How to rely the successive hash choices for successive non-deterministic choices?

Proposition:
- Initialize with a random seed (initial state)
- Linear congruential method \( r_{n+1} = (a \cdot r_n + c) \mod m \) for successive states
- \( m \) is the maximum number of different hash-codes
- A pseudo-random number generator maps a value to a pseudo-statistically independent value

Problem:
- Does not account for history and does not differentiate states
Hash for Memoryless scheduler algorithm

- Hash the seed with state’s variables.

**Problem:** Does not take history into account.

**Solution:** Hash the history of states.
Smart sampling

Problem: Simple sampling assigns equal budget to all schedulers

Solution: iteratively refine a candidate set of schedulers
- initial candidates chosen to maximise probability of being optimal
- iterate: estimate, reject worst half, re-assigning budget to remainder
- as the set is reduced, the confidence increases.

Question: How to select the set of initial candidates?
Compute an initial set of candidates

- Coarsely estimate:
  - probability of property under optimal scheduler ($p_s$)
  - probability of finding a near-optimal scheduler ($p_g$)

- Maximise $(1 - (1 - p_g)^M)(1 - (1 - p_s)^N)$:
  - Start from initial budget $NM = N_{\text{max}}$
  - Identify the scheduler with the best probability $P_s$
  - Allows us to get an estimation of $p_g$ and $p_s$
  - Use it to maximize $N$ and $M$. Here: $N = 1/p_s$
Generating the initial candidate set

With fixed budget, naive sampling strategies reduce probability of seeing good scheduler

Smart sampling maximises prob. of seeing good scheduler

too many schedulers too few schedulers optimal number of schedulers

⇒ large areas of very low probability ⇒ large area of high probability
Outline

1. Introduction
2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work
3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work
4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work
5. Conclusion
Simple example

\[ X(\psi \land XG_{\leq 4} \neg \psi) \]

- It is satisfied by \( s_0 s_1 s_0 s_0 \cdots \)
- If \( p_1 > p_2 \), the maximum probability for \( s_0 s_1 \) is achieved with \( a_2 \), while the maximum probability for \( s_0 s_0 \) is achieved with \( a_1 \).
Observation

- Evolution of probability of collision in time.
- Dashed line: proportion of schedulers with probability zero, decreases with time for collision (cannot be obtained with numerical model checking)
- Smart sampling is 200 times faster.
Outline

1. Introduction
2. Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work
3. Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work
4. Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work
5. Conclusion
Future Work

- Much faster than PRISM
- Existing work (Peyronnet, Clarke) try to remove non-determinism or improve rewards
- Work of Class is based on local action/states, hence only memoryless
- Near-optimal schedulers may be very rare
- Statistical confidence does not guarantee optimality
- Sampling considers only subset of schedulers
- Near-optimal schedulers may be rare
- Smart sampling only improves efficiency
- Comparison with learning-based techniques
- Computing rewards.
Outline

1 Introduction
2 Plasma: a social application
   - Architecture of the tool
   - Application
   - Future Work
3 Importance Splitting
   - Principles
   - Contributions
   - Evaluation
   - Future Work
4 Smart Sampling
   - Principles
   - Contribution
   - Evaluation
   - Future Work
5 Conclusion
Conclusion

Contributions:

- A new modular tool for the evaluation of SMC algorithms
- New application domains for SMC
- CUSUM, timed systems, adaptive systems (not presented today)
- Industry impact and press coverage.
Future work

- Combining the algorithms developed in this presentation
  - rare events for timed systems
  - non-determinism and rare events.

- Applications to Systems of Systems and internet of things

- Applications to security issues
  - A model checker for vulnerability analysis (at assembly level)
  - How to account for non-determinism introduced by static analysis of assembly instruction?
  - How to represent malware signatures with logical formulae?
  - How to detect rare modification of the code?
  - How to model hardware faults via (stochastic) transition systems?