Performability Measure Specification: Combining CSRL and MSL (ongoing work)

Alessandro Aldini[1], Marco Bernardo[1], Jeremy Sproston[2]

[1]Dipartimento di Matematica, Fisica e Informatica, University of Urbino "Carlo Bo", Italy

[2]Dipartimento di Informatica, University of Turin, Italy

L'Aquila, 2nd March 2010

Motivation

- Aim: further develop methods for the specification of performability measures.
- Define UMSL: a unified measure specification language.
- Combine two proposals made previously:
  - Stochastic temporal logic: CSRL [BaierHaverkortHermannsKatoen00].
  - Component-oriented language based on first-order logic: MSL [AldiniBernardo07].

Motivation

- CSRL is used to specify *path-based properties* of stochastic systems.
    - For example: "with probability at most 0.02, the system exhibits an execution path such that:
        - there is a point along the path at which the atomic proposition *fail* holds directly after performing a *break* action;
        - at this point no more than 30 time units have elapsed and at most 7 units of reward have been accumulated,
        - *up* holds at all preceding points,
        - and the action *ok* is the only action seen on the path before the action *break*."

Motivation

- MSL is used to specify *reward structures* of stochastic systems.
  - Reward structure: assign a real number to each state corresponding to reward/cost per time unit accumulated in that state.
  - Component-oriented:
    - System state: vector of local states, one for each component.
    - MSL uses first-order logic to characterize the assignment of rewards to states.
    - Statements of the logic are built from local states and/or local activities (actions).

Motivation

- What is missing?
  - MSL can handle classical measures (throughput, utilization, response time etc.), but not path-based properties.
  - CSRL assumes the presence of a reward structure; its atomic propositions are not (necessarily) component-ortiented.

Reference model

- Finite labeled CTMC: $\mathcal{M} = (S, T; L_{\mathrm{s}}, L_{\mathrm{t}}; N, Loc, Act)$
  - S is a finite set of states.
  - $T \subseteq S \times \mathbb{R}_{>0} \times S$ is a finitely branching transition relation.
  - $L_{\mathrm{s}} : S \rightarrow Loc^N$ is a labeling function for states.
  - $L_{\mathrm{t}} : T \rightarrow Act$ is a labeling function for transitions.
- Use $s \xrightarrow{a,\lambda}_{\mathcal{M}} s'$ to denote a labeled transition.
- Use $[z_1, z_2, \ldots, z_N]$ to denote vectors of local states (which label CTMC states).
- $z \in s$ (resp. $z \notin s$) denotes that $z = z_i$ for some $1 \leq i \leq N$ (resp. $z \neq z_i$ for all $1 \leq i \leq N$).

Reference model

- Example: system with two identical servers.
  - Requests arrive at the system with rate $\lambda \in \mathbb{R}_{>0}$.
  - When a request finds both servers busy, it must immediately leave the system.
  - When a request finds both servers idle, it has the same probability to be accepted by the two servers.
  - For $i \in \{1, 2\}$, server i processes requests with rate μ_i, fails with rate φ_i, is repaired with rate ϱ_i.
- In Markovian process algebra:
  - Arrival process A: $\mathit{Arrivals} \stackrel{\Delta}{=} <\mathit{arrive}, \lambda>.\mathit{Arrivals}$
  - Server S_i:

$$\begin{aligned}
\mathit{Idle} &\stackrel{\Delta}{=} <\mathit{arrive}, *>.\mathit{Busy} \\
\mathit{Busy} &\stackrel{\Delta}{=} <\mathit{serve}, \mu_i>.\mathit{Idle} \; + \; <\mathit{fail}, \varphi_i>.\mathit{Failed} \\
\mathit{Failed} &\stackrel{\Delta}{=} <\mathit{repair}, \varrho_i>.\mathit{Busy}
\end{aligned}$$

- A has a single local state: $A.\mathit{Arrivals}$.
- S_i has three local states: $S_i.\mathit{Idle}$, $S_i.\mathit{Busy}$, and $S_i.\mathit{Failed}$.

Formulas on local states and activities

- Define *dnfLoc*: disjunctive normal forms on local states

$$Z = (z_{1,1} \wedge \ldots \wedge z_{1,m_1}) \vee \ldots \vee (z_{n,1} \wedge \ldots \wedge z_{n,m_n})$$

  where each literal $z_{i,j}$ is either a local state or the negation of a local state.

- Define *dnfAct*: disjunctive normal forms on activities

$$A = (a_{1,1} \wedge \ldots \wedge a_{1,m_1}) \vee \ldots \vee (a_{n,1} \wedge \ldots \wedge a_{n,m_n})$$

  where each literal $a_{i,j}$ is either an activity or the negation of an activity.

- Shorthand: write Z_i for a disjunct of Z (same for A_i and A).

Formulas on local states and activities

- Given state $s \in S$, local state $z \in Loc$, and activity $a \in Act$, define predicate *sat* by letting:

$$
\begin{array}{lll}
s\ sat\ z & \text{iff} & z \in s \\
s\ sat\ \bar{z} & \text{iff} & z \notin s \\
s\ sat\ a & \text{iff} & \exists \lambda \in \mathbb{R}_{>0}, s' \in S.\, s \xrightarrow{a,\lambda}_{\mathcal{M}} s' \\
s\ sat\ \bar{a} & \text{iff} & \nexists \lambda \in \mathbb{R}_{>0}, s' \in S.\, s \xrightarrow{a,\lambda}_{\mathcal{M}} s'
\end{array}
$$

- Then extend *sat* to literal conjunction and disjunction in the expected way.

dnfMSL (preliminaries)

- $eq : \mathbb{R} \times \mathbb{R} \rightarrow \{\mathrm{true}, \mathrm{false}\}$ such that:

$$eq(x, y) \;=\; \begin{cases} \mathrm{true} & \text{if } x = y \\ \mathrm{false} & \text{otherwise} \end{cases}$$

- $state\_rew : S \rightarrow \mathbb{R}$ such that $state\_rew(s)$ is the reward accumulated while staying in state s due to either local states of s or activities enabled by s.
- $lstate\_rew : Loc \cup negLoc \rightarrow \mathbb{R}$ such that $lstate\_rew(z)$ is the reward contribution given by local state z and $lstate\_rew(\bar{z})$ is the reward contribution given by the negation of local state $\bar{z}$.
- $act\_rew : Act \cup negAct \rightarrow \mathbb{R}$ such that $act\_rew(a)$ is the reward contribution given by activity a and $act\_rew(\bar{a})$ is the reward contribution given by the negation of activity $\bar{a}$.

dnfMSL (1)

$$\boxed{eq(state\_rew(s), sum\_lstate\_contrib(s, Z, af))}$$

where:

- $sum\_lstate\_contrib : S \times dnfLoc \times AF \rightarrow \mathbb{R}$ such that:

$$sum\_lstate\_contrib(s, Z, af) = \sum_{Z_i \in Z \text{ s.t. } s\ sat\ Z_i} af \{\!|\ lstate\_rew(z) \mid z \text{ occurs in } Z_i\ |\!\}$$

- $af : 2^{\mathbb{R}} \rightarrow \mathbb{R}$: arithmetical function, e.g., sum, min, max, or average.

dnfMSL (2)

$$\boxed{eq(state\_rew(s), sum\_act\_contrib(s, A, af))}$$

where:

- $sum\_act\_contrib : S \times dnfAct \times AF \rightarrow \mathbb{R}$ such that:

$$sum\_act\_contrib(s, A, af) = \sum_{A_i \in A \text{ s.t. } s\ sat\ A_i} af\,\{\!|\ act\_rew(a) \mid a \text{ occurs in } A_i\ |\!\}$$

dnfMSL (3)

$$\boxed{eq(state\_rew(s), choose\_lstate\_contrib(s, Z, af, cf))}$$

where:

- $choose\_lstate\_contrib : S \times dnfLoc \times AF \times CF \rightarrow \mathbb{R}$ such that:

$$choose\_lstate\_contrib(s, Z, af, cf) \ = \ \underset{Z_i \in Z \text{ s.t. } s\ sat\ Z_i}{cf} af\{\!| \ lstate\_rew(z) \mid z \text{ occurs in } Z_i \ |\!\}$$

- $cf : 2^{\mathbb{R}} \rightarrow \mathbb{R}$: choice functions; i.e., $cf(\emptyset) = 0$ and $cf(\{x_1, \ldots, x_n\}) \in \{x_1, \ldots, x_n\}$ for all $n \in \mathbb{N}_{>0}$.

dnfMSL (4)

$$\boxed{eq(state\_rew(s), choose\_act\_contrib(s, A, af, cf))}$$

where:

- $choose\_act\_contrib : S \times dnfAct \times AF \times CF \rightarrow \mathbb{R}$ such that:

$$\begin{aligned} &choose\_act\_contrib(s, A, af, cf) \;= \\ &\mathop{cf}_{A_i \in A \text{ s.t. } s\; sat\; A_i} af\, \{\!|\, act\_rew(a) \mid a \text{ occurs in } A_i \,|\!\} \end{aligned}$$

aCSRL: syntax

- State formulas of aCSRL:

$$\boxed{\Phi \ ::= \ Z \mid A \mid \Phi \wedge \Phi \mid \neg\Phi \mid \mathcal{S}_{\bowtie p}(\Phi) \mid \mathcal{P}_{\bowtie p}(\varphi)}$$

  where $Z \in dnfLoc$ is a disjunctive normal form on local states, $A \in dnfAct$ is a disjunctive normal form on activities, $\bowtie \in \{<, \leq, \geq, >\}$ is a comparison operator, $p \in \mathbb{R}_{[0,1]}$ is a probability.

- Path formulas:

$$\boxed{\varphi \ ::= \ \Phi \ {}_{\mathcal{A}}U^{\leq t}_{\leq r} \ \Phi \mid \Phi \ {}_{\mathcal{A}_1}U^{\leq t}_{\leq r} \ {}_{\mathcal{A}_2} \ \Phi}$$

  where $\mathcal{A}_1, \mathcal{A}_2 \subseteq Act$ are sets of actions and $t, r \in \mathbb{R}_{\geq 0}$ are nonnegative reals.

aCSRL: semantics (informally)

- Let $\omega = s_0 \xrightarrow{a_0, t_0} s_1 \xrightarrow{a_1, t_1} \cdots$ be a path (infinite sequence of steps).
- $\omega \models \Phi_1 \, {}_{\mathcal{A}}U^{\leq t}_{\leq r} \, \Phi_2$: the path visits a state satisfying Φ_2 within t time units, while accumulating at most r reward, and visits states satisfying Φ_1 while performing only actions in $\mathcal{A}$ until that point.
- $\omega \models \Phi_1 \, {}_{\mathcal{A}_1}U^{\leq t}_{\leq r} \, {}_{\mathcal{A}_2}\Phi_2$: the path visits a state satisfying Φ_2 within t time units, while accumulating at most r reward, after performing an action in $\mathcal{A}_2$, and visits states satisfying Φ_1 while performing only actions in $\mathcal{A}_1$ until that point.

aCSRL: semantics (informally)

- $s \models \mathcal{P}_{\bowtie p}(\varphi)$ iff $Prob_s^{\mathcal{M}}(\{\omega \in Path^{\mathcal{M}} \mid \omega \models \varphi\}) \bowtie p$ where $Prob_s^{\mathcal{M}}(\Omega)$ is the probability of exhibiting paths in Ω from s in $\mathcal{M}$.
- Example: "with probability at most 0.02, the system exhibits an execution path such that:
    - there is a point along the path at which the atomic proposition *C.fail* holds directly after performing a *break* action;
    - at this point no more than 30 time units have elapsed and at most 7 units of reward have been accumulated,
    - *C.up* holds at all preceding points,
    - and the action *ok* is the only action seen on the path before the action *break*."

results in $\mathcal{P}_{\leq 0.02}(C.up\ {}_{\{ok\}}U^{\leq 30}_{\leq 7}\ {}_{\{break\}}\ C.fail)$.

UMSL: combining aCSRL and dnfMSL

- Proposal:
  - Define dnfMSL+:
    a "formula" of dnfMSL+ is a pair (ψ, Φ), where ψ is a dnfMSL formula and Φ is an aCSRL+ formula.
  - Define aCSRL+:
    a "formula" of aCSRL+ is a pair (ϕ, Ψ), where ϕ is an aCSRL formula and Ψ is a dnfMSL+ formula.
  - Can nest aCSRL+ formulas within an aCSRL formula.
- Intuition:
  - dnfMSL+ formula (ψ, Φ):
    defines a reward structure by *applying ψ only to states satisfying* Φ.
  - aCSRL+ formula (ϕ, Ψ):
    is satisfied by the states satisfying ϕ where the reward structure is given by Ψ.
- Details to be worked on...

Measure definition mechanism for UMSL (1)

- Aim: to create a set of high-level of performance measures formed by UMSL.
- Syntax:

$$\texttt{MEASURE} \ \triangleleft \, \textit{name} \triangleright (\triangleleft \textit{parameters} \triangleright) \ \texttt{IS} \ \triangleleft \, \textit{body} \triangleright$$

  where *parameters* refers to component-oriented arguments, and *body* contains the UMSL formula.
- Example: system throughput with respect to the activities $C_1.a_1, \ldots, C_N.a_N$:

$$\texttt{MEASURE} \ \textit{throughput}(C_1.a_1, \ldots, C_N.a_N) \ \texttt{IS} \, (\psi, (\Phi, \textit{true}))$$

  where $\Phi = A$ and ψ is defined by

$$\textit{eq}(\textit{state\_rew}(s), \textit{sum\_act\_contrib}(s, A, \mathrm{sum}))$$

  such that $A = (C_1.a_1 \vee \ldots \vee C_N.a_N)$ and $\textit{act\_rew}(a) = \lambda$ whenever $a = C_i.a_i$ for some $1 \leq i \leq N$ and the rate associated with a is λ.

Measure definition mechanism for UMSL (2)

- Example: states (1) not including the behavior $C.B$ and (2) from which $C.B$ becomes true within t time units with at most reward r, where throughput is accumulated by $C.a$ actions, with probability $\leq p$:

  $$\texttt{MEASURE}\ \mathit{throughput\_until\_beh}(C.B, C.a, p)\ \texttt{IS}\,(\psi, (\Phi, \psi'))$$

  where ψ is defined by

  $$\mathit{eq}(\mathit{state\_rew}_{\Phi}(s), \mathit{choose\_lstate\_contrib}(s, \{\neg C.B\}, \mathrm{sum}, \mathsf{min}))$$

  such that $\mathit{lstate\_rew}(\neg C.B) = 1$, where:

  $$\Phi = (\mathcal{P}_{\leq p}(\neg C.B \ \ {}_{\mathit{Act}}U^{\leq t}_{\leq r} \ \ C.B), \Psi)\,,$$

  and where ψ' is defined by:

  $$\mathit{eq}(\mathit{state\_rew}(s), \mathit{sum\_act\_contrib}(s, \{a\}, \mathrm{sum}))$$

  where $\mathit{act\_rew}_{\mathit{path}}(a) = \lambda$ whenever the rate associated with a is λ.

- $\mathit{state\_rew}_{\Phi}(s)$ is assigned a reward only if $s \models \Phi$, otherwise reward is 0.

Remaining work

- Define formally the syntax and semantics of UMSL.
- Describe the model-checking algorithm of aCSRL (combination of that of aCSL [HKMS00] and CSRL [BHHK00,HCHKB02]).
- Work further on measure definition mechanism (patterns [Grunske09] and nesting).
- Examples.