Symbolic Limited Lookahead Control for Best-effort Dynamic Computing Resource Management

Nicolas Berthier
University of Liverpool

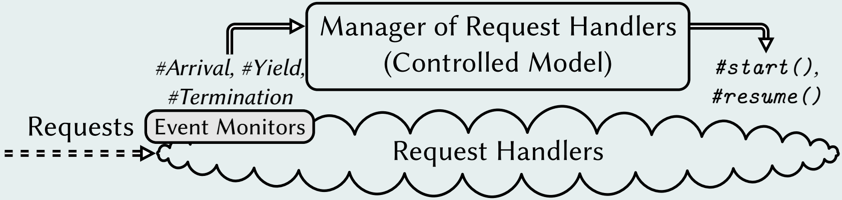
Hervé Marchand
INRIA Rennes
Bretagne Atlantique

Éric Rutten
Université Grenoble Alpes
INRIA, CNRS

WODES 2018 — May 30

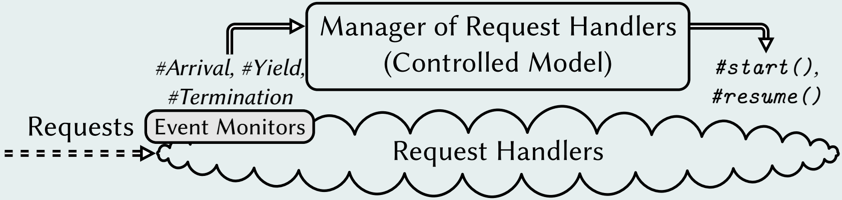
Dynamic Resource Management: Use-case

Model-based Management of Request Handlers

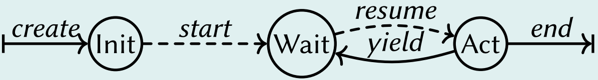


Dynamic Resource Management: Use-case

Model-based Management of Request Handlers

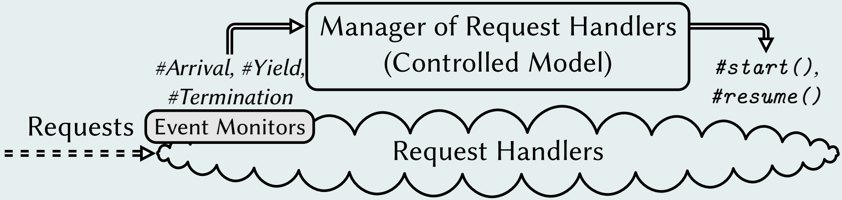


Example (Request Handler Behavior)

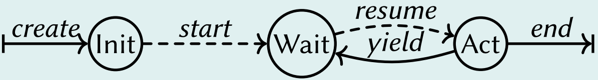


Dynamic Resource Management: Use-case

Model-based Management of Request Handlers



Example (Request Handler Behavior)



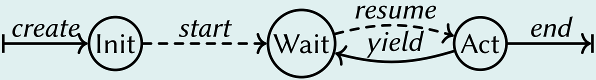
Symbolic Modeling Principle

Location ⇝ Counter state variable

Event / command ⇝ Non-controllable / controllable input variable

Dynamic Resource Management: Linear ASTS Model

Example (Behavior of One Request Handler)



Example (Arithmetic Symbolic Transition System S_{rh})

$$X = \langle init, wait, active \colon \mathbb{Z}^3 \rangle$$

$$U = \langle create, yield, end \colon \mathbb{Z}^3 \rangle$$

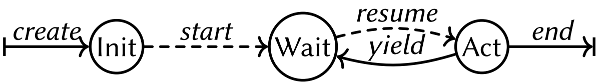
$$C = \langle start, resume \colon \mathbb{Z}^2 \rangle$$

$$T = \begin{cases} init & := init - start + create \\ wait & := wait + start + yield - resume \\ active & := active + resume - yield - end \end{cases}$$

$$A = start \geqslant 0 \wedge resume \geqslant 0 \wedge yield \geqslant 0 \wedge create \geqslant 0 \wedge end \geqslant 0 \wedge$$
$$start \leqslant init \wedge resume \leqslant wait \wedge yield + end \leqslant active$$

$$X_0 = init = 0 \wedge wait = 0 \wedge active = 0$$

Dynamic Resource Management: Discrete Control



Example (Management Goals as Control Objectives for S_{rh})

Restricting the number of started handlers ← *Safety*

- *e.g.*, $wait + active \leqslant 42$

Minimizing the number of non-started handlers ← *Optimization*

- *i.e.*, Minimizing *init*

Symbolic Discrete Controller Synthesis on ASTSs

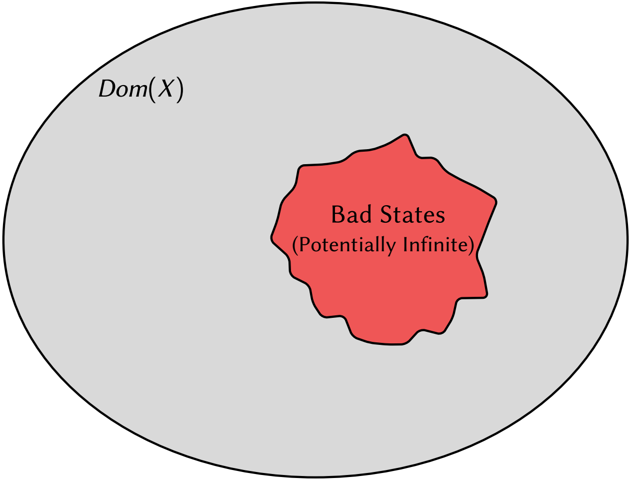
Principles

- Given S and a *Control Objective* o
- Compute a *Controller* $K \in Pred(X \cup U \cup C)$ (a *Predicate* involving variables in X, U, and C) *s.t* $S/_K$ fulfills o

Interpretation / Semantics of $S/_K$

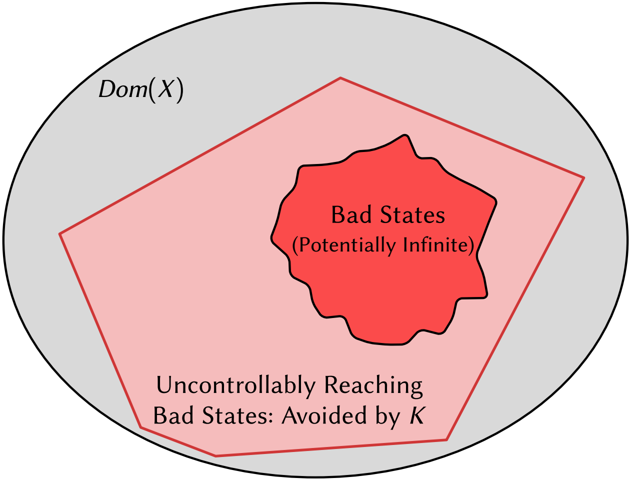
- Start in $q \in Dom(X)$ *s.t* $q \models X_0$
- Repeat
  1. Receive *Admissible* $\upsilon \in Dom(U)$ (*s.t* $\exists \gamma \in Dom(C), (q, \upsilon, \gamma) \models A$)
  2. Choose $\gamma \in Dom(C)$ *s.t* $(q, \upsilon, \gamma) \models K$
  3. $q \leftarrow T(q, \upsilon, \gamma)$
- *Block* at step 1. if $\nexists (\upsilon, \gamma) \in Dom(U) \times Dom(C), (q, \upsilon, \gamma) \models A$
- *Deadlock* at step 2. if $\nexists \gamma \in Dom(C), (q, \upsilon, \gamma) \models K$

Previous Work: Control for Safety[1]



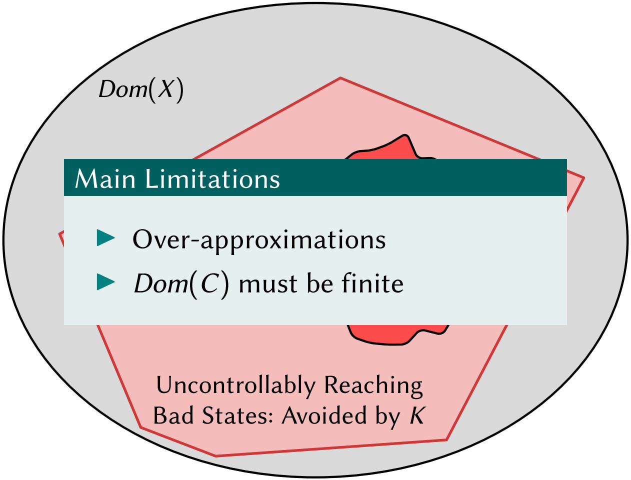
[1] Nicolas Berthier and Hervé Marchand. "Discrete Controller Synthesis for Infinite State Systems with ReaX". In: *12th Int. Workshop on Discrete Event Systems*. WODES '14. Cachan, France: IFAC, May 2014, pp. 46–53; Nicolas Berthier and Hervé Marchand. "Deadlock-Free Discrete Controller Synthesis for Infinite State Systems". In: *54th IEEE Conference on Decision and Control*. CDC '15. Dec. 2015.

Previous Work: Control for Safety[1]



[1] Nicolas Berthier and Hervé Marchand. "Discrete Controller Synthesis for Infinite State Systems with ReaX". In: *12th Int. Workshop on Discrete Event Systems.* WODES '14. Cachan, France: IFAC, May 2014, pp. 46–53; Nicolas Berthier and Hervé Marchand. "Deadlock-Free Discrete Controller Synthesis for Infinite State Systems". In: *54th IEEE Conference on Decision and Control.* CDC '15. Dec. 2015.

Previous Work: Control for Safety[1]



Main Limitations

- Over-approximations
- $Dom(C)$ must be finite

[1] Nicolas Berthier and Hervé Marchand. "Discrete Controller Synthesis for Infinite State Systems with ReaX". In: *12th Int. Workshop on Discrete Event Systems.* WODES '14. Cachan, France: IFAC, May 2014, pp. 46–53; Nicolas Berthier and Hervé Marchand. "Deadlock-Free Discrete Controller Synthesis for Infinite State Systems". In: *54th IEEE Conference on Decision and Control.* CDC '15. Dec. 2015.

Best-effort Control

Best-effort Control

Compute a *Controller* K *s.t* $S/_K$ "*does its best*" to fulfill its objectives

Best-effort Control

Best-effort Control

Compute a *Controller K s.t* $S/_K$ "*does its best*" to fulfill its objectives

Rationale

- Targeting *non-critical* application domains
- Handle infinite $Dom(C)$ (in addition to infinite $Dom(X)$ and $Dom(U)$)

Symbolic κ-Lookahead Control: Overview

1. Given $\kappa \in \mathbb{N}^+$
2. Introduce additional input variables that represent *future inputs* (both controllable and non-controllable) to peek into κ future steps with κ-*lookahead expressions*: $U_2, \ldots, U_\kappa, C_2, \ldots, C_\kappa$

Symbolic κ-Lookahead Control: Overview

1. Given $\kappa \in \mathbb{N}^+$
2. Introduce additional input variables that represent *future inputs* (both controllable and non-controllable) to peek into κ future steps with κ*-lookahead expressions*: $U_2, \ldots, U_\kappa, C_2, \ldots, C_\kappa$
3. Specify *desirable* paths $\mathcal{R}_\kappa$ of length κ using a κ-lookahead predicate
   - belongs to $Pred(X \cup U \cup U_2 \cup \ldots \cup U_\kappa \cup C \cup C_2 \cup \ldots \cup C_\kappa)$
   - $\mathcal{R}_\kappa$ encodes a potentially infinite set of finite paths

Symbolic κ-Lookahead Control: Overview

1. Given $\kappa \in \mathbb{N}^+$
2. Introduce additional input variables that represent *future inputs* (both controllable and non-controllable) to peek into κ future steps with *κ-lookahead expressions*: $U_2, \ldots, U_\kappa, C_2, \ldots, C_\kappa$
3. Specify *desirable* paths $\mathcal{R}_\kappa$ of length κ using a κ-lookahead predicate
   - belongs to $Pred(X \cup U \cup U_2 \cup \ldots \cup U_\kappa \cup C \cup C_2 \cup \ldots \cup C_\kappa)$
   - $\mathcal{R}_\kappa$ encodes a potentially infinite set of finite paths
4. Compute *controllable prefixes* of $\mathcal{R}_\kappa$ iteratively, down to $\mathcal{R}_1$
   - Alternating universal/existential elimination of the U_i's and C_i's
   - $\mathcal{R}_1$ denotes all transitions that guarantee the existence of valuations for controllable variables for the next $\kappa - 1$ subsequent steps such that the system follows a path belonging to $\mathcal{R}_\kappa$
5. Build a *strict* controller K from $\mathcal{R}_1$
   - $S/_K$ deadlocks whenever one cannot choose values for controllable variables to follow a complete path in $\mathcal{R}_\kappa$

Symbolic κ-Lookahead Control: Overview

1. Given $\kappa \in \mathbb{N}^+$
2. Introduce additional input variables that represent *future inputs* (both controllable and non-controllable) to peek into κ future steps with *κ-lookahead expressions*: $U_2, \ldots, U_\kappa, C_2, \ldots, C_\kappa$
3. Specify *desirable* paths $\mathcal{R}_\kappa$ of length κ using a κ-lookahead predicate
   - belongs to $Pred(X \cup U \cup U_2 \cup \ldots \cup U_\kappa \cup C \cup C_2 \cup \ldots \cup C_\kappa)$
   - $\mathcal{R}_\kappa$ encodes a potentially infinite set of finite paths
4. Compute *controllable prefixes* of $\mathcal{R}_\kappa$ iteratively, down to $\mathcal{R}_1$
   - Alternating universal/existential elimination of the U_i's and C_i's
   - $\mathcal{R}_1$ denotes all transitions that guarantee the existence of valuations for controllable variables for the next $\kappa - 1$ subsequent steps such that the system follows a path belonging to $\mathcal{R}_\kappa$
5. Build a *strict* controller K from $\mathcal{R}_1$
   - $S/_K$ deadlocks whenever one cannot choose values for controllable variables to follow a complete path in $\mathcal{R}_\kappa$
6. Transform K for *best-effort* and/or *recovery*

Symbolic κ-Lookahead: Control Objectives

Definition (Desirable Paths Enforcing $\Phi \in \mathit{Pred}(X)$ for κ Steps Ahead)

$$R_\kappa = \Phi\big|_1 \wedge \ldots \wedge \Phi\big|_\kappa$$

Symbolic κ-Lookahead: Control Objectives

Definition (Desirable Paths Enforcing $\Phi \in Pred(X)$ for κ Steps Ahead)

$$R_\kappa = \Phi|_1 \wedge \ldots \wedge \Phi|_\kappa$$

Definition (Desirable Paths Minimizing $e \in GuardLin(X)$ over κ Steps)

- $C_1, \ldots, C_\kappa$ are the future non-controllable inputs (with $C_1 = C$)
- $C'_1, \ldots, C'_\kappa$ encode “alternative” future non-controllable inputs
- $E_\kappa = \sum_{i \in \{1,\ldots,\kappa\}} e|_i$
- $A_\kappa = \bigwedge_{i \in \{1,\ldots,\kappa\}} A|_i$
- $E'_\kappa = E_\kappa[C_1 \cup \ldots \cup C_\kappa / C'_1 \cup \ldots \cup C'_\kappa]$

$$R_\kappa = \nexists_{C'_1 \cup \ldots \cup C'_\kappa} \left(A_\kappa \Rightarrow \left(A'_\kappa \wedge E'_\kappa < E_\kappa\right)\right)$$

Symbolic κ-Lookahead: Controllers

Definition (Strict Controller K_κ for Desirable Paths $\mathcal{R}_\kappa$)

$S/_{K_\kappa}$ deadlocks in a state $q \in Dom(X)$ with non-controllable inputs $\upsilon \in Dom(U)$ unless it can follow a full path belonging to $\mathcal{R}_\kappa$

Symbolic κ-Lookahead: Controllers

Definition (Strict Controller K_κ for Desirable Paths $\mathcal{R}_\kappa$)

$S/_{K_\kappa}$ deadlocks in a state $q \in Dom(X)$ with non-controllable inputs $\upsilon \in Dom(U)$ unless it can follow a full path belonging to $\mathcal{R}_\kappa$

Definition (Best-effort Controller $\mathrm{BestEffort}(K)$ from Controller K for S)

$S/_{\mathrm{BestEffort}(K)}$ behaves as $S/_K$ whenever it does not deadlock, as S otherwise

Symbolic κ-Lookahead: Controllers

Definition (Strict Controller K_κ for Desirable Paths $\mathcal{R}_\kappa$)

$S/_{K_\kappa}$ deadlocks in a state $q \in Dom(X)$ with non-controllable inputs $\upsilon \in Dom(U)$ unless it can follow a full path belonging to $\mathcal{R}_\kappa$

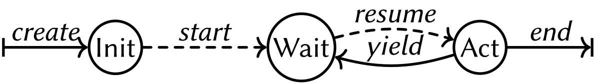
Definition (Best-effort Controller $\mathrm{BestEffort}\,(K)$ from Controller K for S)

$S/_{\mathrm{BestEffort}(K)}$ behaves as $S/_K$ whenever it does not deadlock, as S otherwise

Definition (1-Step Recovering Controller $\mathrm{Recover}\,(K|\mathcal{R}_\kappa)$)

$S/_{\mathrm{Recover}(K|\mathcal{R}_\kappa)}$ behaves as $S/_K$ whenever $S/_{K_\kappa}$ would not deadlock, or transitions in one step to a state where it can follow a full path belonging to $\mathcal{R}_\kappa$ whenever possible

Management Goals as Control Objectives: Examples



Example (Safety Objective)

- Restricting the number of started handlers
- Strict safety control enforcing $\Phi = (wait + active \leqslant 42)$

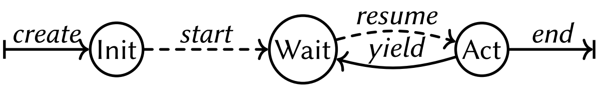
$$\rightsquigarrow K_\Phi$$

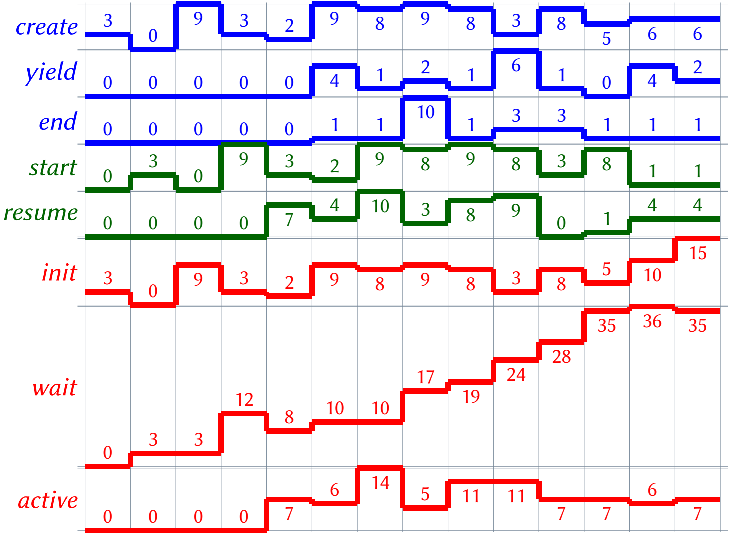
Example (Optimization Objective)

- Minimizing the number of non-started handlers
- Strict minimization of *init* on $S/_{K_\Phi}$ (*i.e.*, with $A = K_\Phi$)

$$\rightsquigarrow K_\omega$$

One Execution Trace of $S_{\mathrm{rh}}/\kappa_\omega$





Experimental Assessment of Practicality (using ReaX)

Benchmarks Derived from the Use-case

Parallel composition: $\|_{i\in\{1,\ldots,N\}} S_{\mathrm{rh}\,i}$ $\rightsquigarrow$ N kinds of request handlers

- Objective: balance the number of active request handlers of each kind

Alt: S_{rh} with N counters $active_i$, $resume_i$, $yield_i$, and end_i
$\rightsquigarrow$ N pools of active request handlers

- Objective: bound the number of non-started request handlers

Table: Synthesis Times (in Seconds)

N:	2	3	6	9
Parallel, κ=1	0.04	0.06	0.28	1.18
Alt, κ=1	0.03	0.04	0.06	0.12
Alt, κ=2	0.10	0.15	0.49	1.76
Alt, κ=1, 1-step recovery	0.05	0.09	0.31	1.73
Alt, κ=2, 1-step recovery	0.14	0.22	0.90	2.02
Alt, κ=3, 1-step recovery	0.27	0.48	1.99	2.78
Alt, κ=4, 1-step recovery	0.53	1.04	4.30	6.87

Related Works

Limited Lookahead

Sheng-Luen Chung, Stephane Lafortune, and Feng Lin. "Limited lookahead policies in supervisory control of discrete event systems". In: *IEEE Trans. Autom. Control* 37.12 (1992), pp. 1921–1935

- Language-theoretic framework, N-step ahead projection of behaviors
- Conservative and optimistic attitudes to accommodate uncertainties

Modeling

Bengt Lennartson et al. "Unified Model for Synthesis and Optimization of Discrete Event and Hybrid Systems". In: *12th Int. Workshop on Discrete Event Systems*. WODES '14. Cachan, France: IFAC, May 2014, pp. 86–92

- Modeling principle similar to ours, though with finite counters (EFAs)
- "Performance" optimization (minimizing the make span of tokens)

Overall Contributions & Future Works

Symbolic Algorithms for Best-effort κ-Lookahead Control

- *Non-critical* safety & optimization
- Control for recovery
- Drop requirement for finite $Dom(C)$
- Implementation available in ReaX[2]

Future Works

- Benchmark with models involving finite variables
- Using MILP or SMT solvers
  - Challenges with universal quantifiers
- Identify “good” models for lookahead

[2] http://reatk.gforge.inria.fr/

Outline

- Symbolic κ-Lookahead
- Return of the Dynamic Resource Management Use-case

Symbolic κ-Lookahead: Peeking into the Future

Representing κ Future Inputs ($\kappa \in \mathbb{N}^+$)

Define new indexed variables $U_2, \ldots, U_\kappa$ and $C_2, \ldots, C_\kappa$ from U and C

- $U_1 = U$, $C_1 = C$, and $I_i = U_i \uplus C_i$

Symbolic κ-Lookahead: Peeking into the Future

Representing κ Future Inputs ($\kappa \in \mathbb{N}^+$)

Define new indexed variables $U_2, \ldots, U_\kappa$ and $C_2, \ldots, C_\kappa$ from U and C

- $U_1 = U$, $C_1 = C$, and $I_i = U_i \uplus C_i$

i-lookahead Expressions ($i \in \{0, \ldots, \kappa\}$)

$e|_i \in Expr(X \cup I_1 \cup \ldots \cup I_i)$ symbolically denotes the (potential) value of $e \in Expr(X)$ i steps ahead

Symbolic κ-Lookahead: Peeking into the Future

Representing κ Future Inputs ($\kappa \in \mathbb{N}^+$)

Define new indexed variables $U_2, \dots, U_\kappa$ and $C_2, \dots, C_\kappa$ from U and C

- $U_1 = U$, $C_1 = C$, and $I_i = U_i \uplus C_i$

i-lookahead Expressions ($i \in \{0, \dots, \kappa\}$)

$e|_i \in \mathit{Expr}(X \cup I_1 \cup \dots \cup I_i)$ symbolically denotes the (potential) value of $e \in \mathit{Expr}(X)$ i steps ahead

Example (2-lookahead of $e \in \mathit{GuardLin}(X)$ with $X = \langle x\colon \mathbb{Z}\rangle$ and $I = \langle b\colon \mathbb{B}\rangle$)

With $T = \langle x := \text{ if } x \leqslant 42 \land b \text{ then } x + 1 \text{ else } x\rangle$ and $e = x$:

$$e|_1 = \begin{cases} x + 1 & \text{if } x \leqslant 42 \land b \\ x & \text{otherwise} \end{cases}$$

$$e|_2 = \begin{cases} x + 2 & \text{if } x \leqslant 41 \land b \land b_2 \\ x + 1 & \text{if } x \leqslant 42 \land (\neg b \land b_2 \lor b \land \neg b_2) \lor x = 42 \land b \land b_2 \\ x & \text{otherwise} \end{cases}$$

Symbolic κ-Lookahead: Control over Sliding Windows

Sliding Windows as i-paths $\mathcal{R}_i$ ($i \in \{0, \ldots, \kappa\}$)

$\mathcal{R}_i \subseteq Dom(X) \times Dom(I)^i$: Set of paths of i transitions

- *Desirable* κ-paths $\mathcal{R}_\kappa$ given as $R_\kappa \in Pred(X \cup I_1 \cup \ldots \cup I_\kappa)$

Symbolic κ-Lookahead: Control over Sliding Windows

Sliding Windows as i-paths $\mathcal{R}_i$ ($i \in \{0, \ldots, \kappa\}$)

$\mathcal{R}_i \subseteq \mathit{Dom}(X) \times \mathit{Dom}(I)^i$: Set of paths of i transitions

- *Desirable* κ-paths $\mathcal{R}_\kappa$ given as $R_\kappa \in \mathit{Pred}(X \cup I_1 \cup \ldots \cup I_\kappa)$

Direct Controllable Prefixes $\mathcal{R}_i$ of $\mathcal{R}_{i+1}$ ($i \in \{0, \ldots, \kappa\}$)

The *direct controllable prefixes* of $(i+1)$-paths $\mathcal{R}_{i+1} \subseteq \mathit{Dom}(X) \times \mathit{Dom}(I)^{i+1}$ consist of all i-paths $\mathcal{R}_i \subseteq \mathit{Dom}(X) \times \mathit{Dom}(I)^i$ such that, after following an i-path in $\mathcal{R}_i$, a valuation for controllable variables always exists so that S remains on an $(i+1)$-path belonging to $\mathcal{R}_{i+1}$

- $\mathrm{prefix}_c^i(R_{i+1}) \stackrel{\text{def}}{=} \forall_{U_i}\left((\exists_{C_i} A|_{i+1}) \Rightarrow \exists_{C_i}(A|_{i+1} \wedge R_{i+1})\right)$
  where $A|_i \in \mathit{Pred}(X \cup I_1 \cup \ldots \cup I_i)$ denotes the set of all i-paths with *admissible* ith transitions

Symbolic κ-Lookahead: Building Controllers

Definition (Strict Controller K_κ for Desirable κ-paths $\mathcal{R}_\kappa$)

$S/_{K_\kappa}$ deadlocks unless it can follow a full κ-path belonging to $\mathcal{R}_\kappa$

- $K_\kappa = A \wedge \mathrm{prefix}_\mathrm{c}^1 \circ \ldots \circ \mathrm{prefix}_\mathrm{c}^{\kappa-1}(R_\kappa)$

Symbolic κ-Lookahead: Building Controllers

Definition (Strict Controller K_κ for Desirable κ-paths $\mathcal{R}_\kappa$)

$S/_{K_\kappa}$ deadlocks unless it can follow a full κ-path belonging to $\mathcal{R}_\kappa$

- $K_\kappa = A \wedge \mathrm{prefix}_c^1 \circ \ldots \circ \mathrm{prefix}_c^{\kappa-1}(R_\kappa)$

Definition (Best-effort Controller $\mathrm{BestEffort}(K)$ from Controller K for S)

$S/_{\mathrm{BestEffort}(K)}$ behaves as $S/_K$ whenever it does not deadlock, as S otherwise

- $\mathrm{BestEffort}(K) = A \wedge (\exists_C K \Rightarrow K)$

Symbolic κ-Lookahead: Building Controllers

Definition (Strict Controller K_κ for Desirable κ-paths $\mathcal{R}_\kappa$)

$S/_{K_\kappa}$ deadlocks unless it can follow a full κ-path belonging to $\mathcal{R}_\kappa$

- $K_\kappa = A \wedge \mathrm{prefix}_c^1 \circ \ldots \circ \mathrm{prefix}_c^{\kappa-1}(R_\kappa)$

Definition (Best-effort Controller $\mathrm{BestEffort}(K)$ from Controller K for S)

$S/_{\mathrm{BestEffort}(K)}$ behaves as $S/_K$ whenever it does not deadlock, as S otherwise

- $\mathrm{BestEffort}(K) = A \wedge (\exists_C K \Rightarrow K)$

Definition (1-Step Recovering Controller $\mathrm{Recover}(K|\mathcal{R}_\kappa)$)

$S/_{\mathrm{Recover}(K|\mathcal{R}_\kappa)}$ behaves as $S/_K$ whenever $S/_{K_\kappa}$ would not deadlock, or reaches a state where it can follow a full κ-path belonging to $\mathcal{R}_\kappa$ whenever reachable in one step

- $\mathrm{Recover}(K|\mathcal{R}_\kappa) = (\nexists_I E \wedge K) \vee (\exists_U E \wedge A)$
  with $E = A \wedge \neg\mathrm{prefix}_c^0(K_\kappa) \wedge \mathrm{prefix}_c^0(K_\kappa)|_1$

Symbolic κ-Lookahead: Optimization Example

Example (Minimizing $E_2 = \sum_{i\in\{1,2\}} x|_i$)

- $X = \langle x\colon \mathbb{Z}\rangle$
- $U = \varnothing$, $C = \langle b\colon \mathbb{B}\rangle$
- $T = \langle x := x + 1 \text{ if } x \leqslant 42 \wedge b, x \text{ otherwise}\rangle$
- $A = \text{tt}$

$$E_2 = \begin{cases} 2x+3 & \text{if } x \leqslant 41 \wedge b \wedge b_2 \\ 2x+2 & \text{if } x \leqslant 42 \wedge b \wedge \neg b_2 \vee x = 42 \wedge b \wedge b_2 \\ 2x+1 & \text{if } x \leqslant 42 \wedge \neg b \wedge b_2 \\ 2x & \text{otherwise} \end{cases}$$

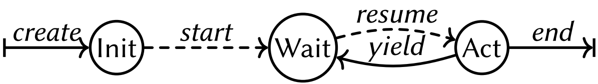
$$R_2 = x \geqslant 43 \vee (\neg b \wedge \neg b_2)$$

$$K_2 = \mathrm{prefix}_c^1(R_2) = (x \leqslant 42 \Rightarrow \neg b)$$

Outline

- Symbolic κ-Lookahead
- Return of the Dynamic Resource Management Use-case

Management Goals as Control Objectives: Safety

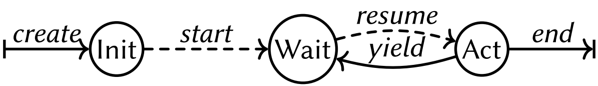


Example (Safety Objective)

- Restricting the number of started handlers
- Strict safety control enforcing $\Phi = (wait + active \leqslant 42)$ with κ=2:

$$
\begin{aligned}
R_2 &= \bigwedge_{i\in\{1,2\}} (wait + active \leqslant 42)|_i \\
&= (wait + active + start - end \leqslant 42)\wedge \\
&\quad (wait + active + start + start_2 - end - end_2 \leqslant 42) \\
K_{\Phi,2} &= A \wedge \mathrm{prefix}^1_c(R_2) \\
&= A \wedge start + wait + active - end \leqslant 42
\end{aligned}
$$

Management Goals as Control Objectives: Optimization



Example (Optimization Objective)

- Minimizing the number of non-started handlers
- Strict minimization of *init* on $S/_{K_{\Phi,2}}$ (*i.e.*, with $A = K_{\Phi,2}$) with κ=1:

$$E_1 = init|_1 = init - start + create$$

$$K_\omega = K_{\Phi,2} \wedge \begin{pmatrix} (start = init \wedge init + wait + active - end \leqslant 41) \vee \\ (start + wait + active - end = 42) \end{pmatrix}$$