

Bounded Multipushdown Systems

Complexity analysis of temporal model-checking problems

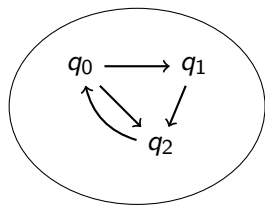
Kshitij Bansal¹, Stéphane Demri^{1,2}

¹New York University ²CNRS

June 29, 2013

CSR, Yekaterinburg, Russia

Pushdown systems

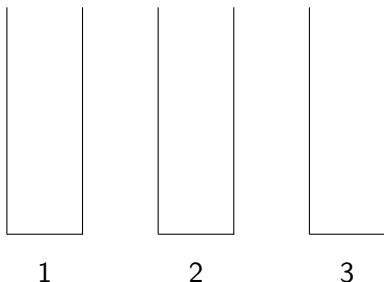
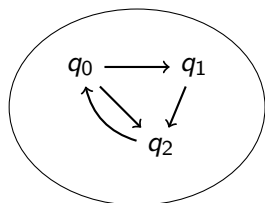


$P = (Q, N, \Gamma, \Delta)$, such that:

- ▶ Q is a non-empty finite set of *states*,
- ▶ Γ is the finite *stack alphabet*
- ▶ Δ is the *transition relation*.

E.g. $q_0 \xrightarrow{(a, \text{push}(b))} q_1 \in \Delta$

Multipushdown systems



$P = (Q, N, \Gamma, \Delta_1, \dots, \Delta_N)$, for some $N \geq 1$ such that:

- ▶ Q is a non-empty finite set of *states*,
- ▶ Γ is the finite *stack alphabet*
- ▶ for every $s \in [N]$, Δ_s is the *transition relation acting on the s -th stack*.

E.g. $q_0 \xrightarrow{(a, \text{push}(b))} q_1 \in \Delta_1$

Multipushdown systems: runs

Configuration: $c = (q, w_1, w_2, \dots, w_N)$.

Run: Sequence of configurations: $c^0 \rightarrow c^1 \rightarrow \dots$ such that consecutive configurations respect transition relation.

Example of run.

q_1		q_2		q_1		q_1		q_1		q_1		q_1		q_1
ϵ	\rightarrow_1	a	\rightarrow_2	a	\rightarrow_1	ab	\rightarrow_1	aa	\rightarrow_2	aa	\rightarrow_1	a	\rightarrow_1	ϵ
ϵ		ϵ		a		a		a		b		b		b

Natural for modeling stack behavior of concurrent programs.

What do we know?

Problem (REACH)

input: *a multipushdown system, an initial state q_0 and final configuration c*

question: *Is there a run from $(q_0, (\epsilon)^N) \rightarrow^* c$?*

Theorem

[Minsky'67]

With just two stacks, the reachability problem is undecidable.

Theorem

e.g. [Schwoon '02]

With one stack, polynomial time procedure for reachability.

This talk

Restrictions of multipushdown systems in other directions to regain decidability and understand complexity.

Restrictions on runs

Divide runs into “phases”. Ask if there is a run with finite number of “phases” that satisfy.

- ▶ Context bounded (B-REACH): phase \equiv all operations only on one stack

$$\begin{bmatrix} q_1 \\ \epsilon \\ \epsilon \end{bmatrix} \xrightarrow{1} \begin{bmatrix} q_2 \\ a \\ \epsilon \end{bmatrix} \xrightarrow{2} \begin{bmatrix} q_1 & q_1 \\ a & ab \\ a & a \end{bmatrix} \xrightarrow{1} \begin{bmatrix} q_1 \\ aa \\ a \end{bmatrix} \xrightarrow{2} \begin{bmatrix} q_1 & q_1 & q_1 \\ aa & a & \epsilon \dots \\ b & b & b \end{bmatrix}$$

Restrictions on runs

Divide runs into “phases”. Ask if there is a run with finite number of “phases” that satisfy.

- ▶ Context bounded (B-REACH): phase \equiv all operations only on one stack

$$\begin{bmatrix} q_1 \\ \epsilon \\ \epsilon \end{bmatrix} \xrightarrow{1} \begin{bmatrix} q_2 \\ a \\ \epsilon \end{bmatrix} \xrightarrow{2} \begin{bmatrix} q_1 & q_1 \\ a & ab \\ a & a \end{bmatrix} \xrightarrow{1} \begin{bmatrix} q_1 \\ aa \\ a \end{bmatrix} \xrightarrow{2} \begin{bmatrix} q_1 & q_1 & q_1 \\ aa & a & \epsilon \dots \\ b & b & b \end{bmatrix}$$

- ▶ Phase bounded (PB-REACH): phase \equiv all pop operations on one stack
- ▶ Ordered pushdown systems (OB-REACH): operations only on “first” non-empty stack

Note: runs can be infinite, more general than considering bound on *length* of runs

More expressive properties of runs

Problem (Repeated reachability)

input: (P, q_0, F) : P multi-pushdown system, $q_0 \in Q$ initial state, and F is a Büchi acceptance set.

question: \exists infinite run ρ from $(q_0, (\epsilon)^N)$ such that a $q_f \in F$ is repeated infinitely often?

More expressive properties of runs

Problem (Repeated reachability)

input: (P, q_0, F) : P multi-pushdown system, $q_0 \in Q$ initial state, and F is a Büchi acceptance set.

question: \exists infinite run ρ from $(q_0, (\epsilon)^N)$ such that a $q_f \in F$ is repeated infinitely often?

Problem (Model checking)

input: (P, q_0, ϕ) where P is a multi-pushdown system, q_0 initial state, ϕ a formula specifying property on runs,

question: \exists infinite run ρ from $(q_0, (\epsilon)^N)$ such that $\rho \models \phi$?

What kind of properties?

Properties

- ▶ Current state?
- ▶ Current thread?
- ▶ Current stack operation – push, pop etc.

What kind of properties?

Properties

- ▶ Current state?
- ▶ Current thread?
- ▶ Current stack operation – push, pop etc.
- ▶ Current stack contents? In particular, we consider *regularity constraints*. [Esparza, Kučera, S. Schwoon '01]

What kind of properties?

Properties

- ▶ Current state?
- ▶ Current thread?
- ▶ Current stack operation – push, pop etc.
- ▶ Current stack contents? In particular, we consider *regularity constraints*. [Esparza, Kučera, S. Schwoon '01]
(given automata \mathcal{A} over Σ and stack s : is the content of stack s accepted by \mathcal{A}).

What kind of properties?

Properties

- ▶ Current state?
- ▶ Current thread?
- ▶ Current stack operation – push, pop etc.
- ▶ Current stack contents? In particular, we consider *regularity constraints*. [Esparza, Kučera, S. Schwoon '01]
(given automata \mathcal{A} over Σ and stack s : is the content of stack s accepted by \mathcal{A}).

When in the run?

- ▶ LTL operators: next, until, finally etc.
- ▶ Call and returns: “when the function returns”, “the function that called”. [Alur, Etessami, Madhusudan '04]
 - ▶ **parameterized by stack.**

Multi-CaRet^{reg} : A logic of calls and returns for multiple stacks with regularity constraints

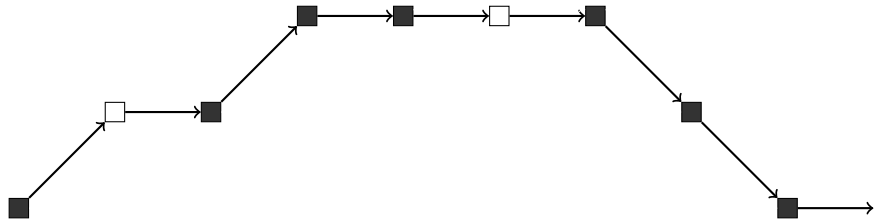
$$\begin{aligned} \phi \quad := \quad & q \mid s \mid \text{in}(s, \mathcal{A}) \mid \text{call} \mid \text{return} \mid \text{internal} \\ & \mid X\phi \mid \phi U \phi \mid X_s^a \phi \mid \phi U_s^a \phi \mid X_s^c \phi \mid \phi U_s^c \phi \\ & \mid \phi \vee \phi \mid \neg \phi \end{aligned}$$

where $s \in [N]$, $q \in Q$.

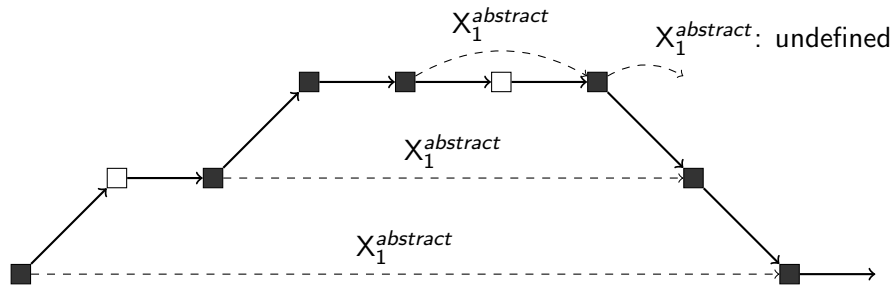
$\rho, t \models q$	iff state at position t is q
$\rho, t \models s$	iff active thread at t is s
$\rho, t \models \text{in}(s, \mathcal{A})$	iff $w_s^t \in L(\mathcal{A})$
$\rho, t \models X_s^{\text{abstract}} \phi$	iff $\rho, \text{Next}_\rho^{\text{abstract}, s}(t) \models \phi$

$\text{Next}_\rho^{\text{abstract}, s}(t)$ is defined as next position when the same stack active within the *same call*.

Example

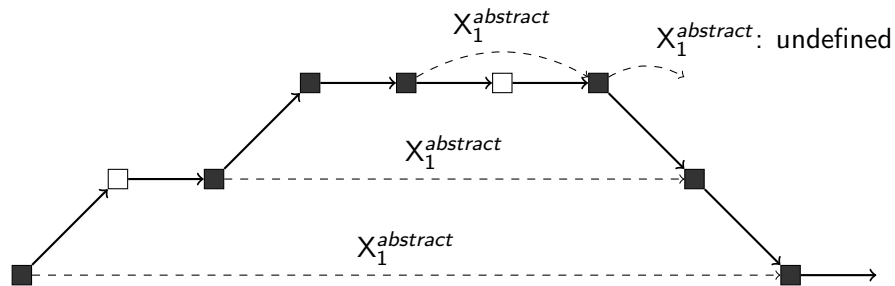


Example



$$X_1^{abstract} \text{in}(1, \mathcal{A}) \implies \text{in}(1, \mathcal{A})$$

Example



$$X_1^{abstract} \text{in}(1, \mathcal{A}) \implies \text{in}(1, \mathcal{A})$$

Can be used to express other related operators, like:

- ▶ $X_s \phi \equiv (\neg s U (s \wedge \phi))$,
- ▶ $X^a \phi \equiv (\bigwedge_s (s \Rightarrow X_s^a \phi))$,
- ▶ $\phi_1 U^a \phi_2 \equiv (\bigwedge_s (s \Rightarrow \phi_1 U_s^a \phi_2))$.

What was known?

	Reach	Rep	LTL	(Multi)CaRet
One stack	P	P	EXPTIME -complete	EXPTIME -complete

What is now known?

	Reach	Rep	LTL	(Multi)CaRet
One stack	P	P	EXPTIME -complete	EXPTIME -complete
k-context* bounded	NP -comp	EXPTIME	EXPTIME	EXPTIME (<i>this paper</i>)

* k part of input, encoded in unary

What is now known?

	Reach	Rep	LTL	(Multi)CaRet
One stack	P	P	EXPTIME -complete	EXPTIME -complete
k-context* bounded	NP -comp	EXPTIME	EXPTIME	EXPTIME (<i>this paper</i>)

Step 2: Arrow from Rep (One stack) to Rep (k-context* bounded)

Step 1: Arrow from (Multi)CaRet (k-context* bounded) to Rep (k-context* bounded)

* k part of input, encoded in unary

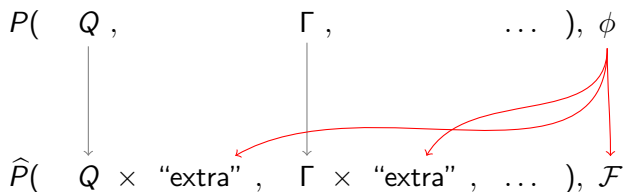
Step 1: From Model-checking to Repeated reachability

$$P(Q, \Gamma, \dots), \phi$$

$$\hat{P}(\hat{Q}, \hat{\Gamma}, \dots), \mathcal{F}$$

P has a run satisfying ϕ iff \hat{P} has a run with states in \mathcal{F} infinitely often. (Büchi acceptance)

Step 1: From Model-checking to Repeated reachability

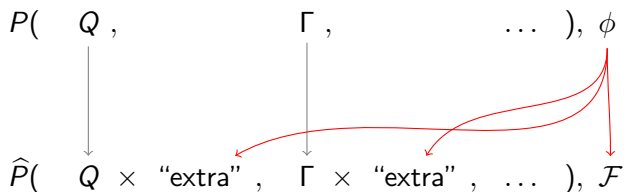


P has a run satisfying ϕ iff \hat{P} has a run with states in \mathcal{F} infinitely often. (Büchi acceptance)

How?

- ▶ Simulate original system
- ▶ Keep extra information to verify satisfaction of formula.

Step 1: From Model-checking to Repeated reachability



P has a run satisfying ϕ iff \hat{P} has a run with states in \mathcal{F} infinitely often. (Büchi acceptance)

How?

- ▶ Simulate original system
- ▶ Keep extra information to verify satisfaction of formula.

What all information?

Step 1: From MC to REP: **high-level details**

In particular we track,

1. In state, about truth value of each subformula (Fischer-Ladner closure for LTL): but **for each stack separately**.
2. In stack, information about call and returns (CaRet).
3. In state, **whether stack will ever be active again**.
4. In state and stack, **automata for satisfaction of regularity constraint**.

Step 1: From MC to REP: **Upshot**

$MC(P, \phi)$

size: $|P| + |\phi|$

$\exists \rho \models \phi$

\iff

$REP(\hat{P}, \mathcal{F})$

size: $O(2^{P(|P|, |\phi|)})$

\exists accepting $\hat{\rho}$

Operation on stacks, stack order as well as actions are preserved:

k -bounded

\iff

k -bounded

k -phase bounded

\iff

k -phase-bounded

\preceq -bounded

\iff

\preceq -bounded

Step 2: Bounded repeated reachability: from N to 1 stacks

Lemma

BREP can be solved in time $\mathcal{O}(|P|^{k+1} \times p(k, |P|))$ for some polynomial $p(\cdot, \cdot)$.

At heart of complexity analysis of this step, two main ideas:

- ▶ For a pushdown system P , post^* (regular set of configurations) is regular. **Also**: can be done polynomial time, size of new automata only additive in $|P|$. [Bouajjani, Esparza, Maler '97]
- ▶ Repeated reachability for 1-stack is in P .

[Boujanni, Esparza, Maler '97]

Connect above two ideas by guessing context-switches and intermediate states.

Other restrictions





	Rep Reach	Multi-CaRetMC
One stack	P	EXPTIME (CaRet)
k-Context bounded*	EXPTIME -complete	EXPTIME -complete
k-Phase bounded*	2-EXPTIME	2-EXPTIME
Order-bounded	2-EXPTIME (2ETIME-hard)	2-EXPTIME (2ETIME-hard)

* k part of input, encoded in unary. For binary encoding, 2EXPTIME (resp. 3EXPTIME) upper bound for context-bounded (resp. phase-bounded) MC.

Step 1: same analysis

Step 2: [Atig, Bollig, Habermehl '08] for PBMC, [Atig '10] for OBMP.

Related work

-  M. Atig, A. Bouajjani, K. N. Kumar, and P. Saivasan.
Linear-time model-checking for multithreaded programs under scope-bounding.
In *ATVA'12*, volume 7561 of *LNCS*, Springer.
-  S. La Torre and M. Napoli.
A temporal logic for multi-threaded programs.
In *TCS'12*, volume 7604 of *LNCS*, Springer.
-  P. Madhusudan and G. Parlato.
The tree width of auxiliary storage.
In *POPL'11*, pages 283–294. ACM.
-  B. Bollig, D. Kuske, and R. Mennicke.
The complexity of model-checking multi-stack systems.
In *LICS'13*.

Conclusion

- ▶ EXPTIME bound for k -bounded runs, Multi-CaRet operators, regularity constraints on stacks. Matches hardness result for single-stack LTL model checking.
- ▶ Approaches to unification
 - ▶ Classes of notions of boundedness (e.g., tree-width bounded).
 - ▶ Classes of logical operators (e.g., specifiable using classes of MSO formulas).