The Complexity of Bounded Context Switching with Dynamic Thread Creation

Pascal Baumann, Rupak Majumdar, Ramanathan S. Thinniyam, Georg Zetzsche

Max Planck Institute for Software Systems (MPI-SWS)

HIGHLIGHTS 2020 (originally ICALP 2020)
Dynamic Networks of Concurrent Pushdown Systems (DCPS)

Model Features:
Dynamic Networks of Concurrent Pushdown Systems (DCPS)

Model Features:
- Concurrent threads with local recursion.
Dynamic Networks of Concurrent Pushdown Systems (DCPS)

Model Features:

- Concurrent threads with local recursion.
- A finite global memory, accessible by all threads.
Dynamic Networks of Concurrent Pushdown Systems (DCPS)

Model Features:
- Concurrent threads with local recursion.
- A finite global memory, accessible by all threads.
- New threads being spawned dynamically during execution.

\[
\begin{array}{c}
  a \\
  a \\
  c
\end{array}
\quad
\begin{array}{c}
  b \\
  b \\
  c
\end{array}
\quad
\begin{array}{c}
  b \\
  b
\end{array}
\]
Dynamic Networks of Concurrent Pushdown Systems (DCPS)

Model Features:

- Concurrent threads with local recursion.
- A finite global memory, accessible by all threads.
- New threads being spawned dynamically during execution.
- Bound $K$ on context switches per thread (avoids undecidability).
Safety Verification

$K$-bounded state reachability problem for DCPS (SRP[$K$])

**Input** A DCPS $A$ and a global state $g$

**Question** Is $g$ $K$-bounded reachable in $A$?
Safety Verification

\(K\)-bounded state reachability problem for DCPS (SRP[\(K\)])

**Input** A DCPS \(A\) and a global state \(g\)

**Question** Is \(g\) \(K\)-bounded reachable in \(A\)?

SRP[0] is EXPSPACE-complete.
- Shown by Ganty and Majumdar (2012).

SRP[\(K\)] is EXPSPACE-hard and in 2EXPSPACE for every \(K \geq 1\).
- Shown by Atig, Bouajjani, and Qadeer (2009).
Safety Verification

\(K\)-bounded state reachability problem for DCPS (SRP[\(K\)])

**Input** A DCPS \(\mathcal{A}\) and a global state \(g\)

**Question** Is \(g\) \(K\)-bounded reachable in \(\mathcal{A}\)?

SRP[0] is EXPSPACE-complete.
- Shown by Ganty and Majumdar (2012).

SRP[\(K\)] is EXPSPACE-hard and in 2EXPSPACE for every \(K \geq 1\).
- Shown by Atig, Bouajjani, and Qadeer (2009).

**Our main result**

SRP[\(K\)] is 2EXPSPACE-hard for every \(K \geq 1\).
Proof Outline

SRP[1] for DCPS
Proof Outline

Coverability for transducer-defined Petri nets (TDPN)*

New techniques

SRP[1] for DCPS

*new model
Proof Outline

Termination of triple-exponentially bounded counter programs

Adapted Lipton construction

Termination of recursive net programs (RNP)*

Adapted Lipton construction

Coverability for transducer-defined Petri nets (TDPN)*

New techniques

SRP[1] for DCPS

*new model
Thank you for your attention!

Any questions?
Locking Inactive Threads

\[ a_1 \ldots a_l \]

\[ \begin{array}{|c|}
\hline
T \\
\hline
a_1 \\
\hline
\vdots \\
\hline
a_l \\
\hline
1 \\
\hline
\end{array} \]

\[ \cong \]

\[ \begin{array}{|ccc|}
\hline
T & a & b \\
\hline
a & c & c \\
\hline
a & c & c \\
\hline
1 & 1 & 1 \\
\hline
\end{array} \]

\[ \cong \]

\[ \begin{array}{|ccc|}
\hline
T & a & b \\
\hline
1 & c & c \\
\hline
1 & c & c \\
\hline
1 & c & c \\
\hline
\end{array} \]
Lifting to 2EXPSPACE

We used $2^{2^d} = 2^{2^{d-1} \cdot 2} = \left(2^{2^{d-1}}\right)^2 = 2^{2^{d-1} \cdot 2^{d-1}}$.

- This means from one level to the next the bound gets squared.

\[
\underbrace{(\cdots \left(2^2\right)^2 \cdots)}_{2^n\text{-times}} \underbrace{)^2}_{n\text{-times}} = 2^{2^n} = 2^{2^{2^n}}
\]
Ganty and Majumdar (2012) consider threads running to completion.
- We can ensure that threads empty their stack in our model.
- This allows us to use their EXPSPACE-completeness result for $K = 0$.

Atig, Bouajjani, and Qadeer (2009) consider a slightly different DCPS:
- Each thread spawns with its parents cs-number plus 1.
- We can simulate our model in theirs using 2 more context switches.
- Reduces our SRP[$K$] to their SRP[$K + 2$].
- This allows us to use their 2EXPSPACE-membership result.
Succinct Representation via Transducers

Use binary addresses $w = u.v$ for places:
- $u$: Role, i.e. which line, counter, or auxiliary place it is.
- $v$: Binary representation of recursion depth $d$.

Let the size of the RNP be $h$, the number of lines of code.
- Each counter appears in at least one line.
- Each line only needs at most one auxiliary place.
- Thus, the number of possibilities for $u$ is linear in $h$.

Make the transducers distinguish each possible triple (pair) of prefixes $u$:
- Considering triples adds an exponent of 3, still poly in $h$. 

[End of Document]
The recursion depth $d$ changes by at most 1 at a time.

- Transducers have to check for equality or off-by-one on postfixes $v$.
- These checks require space linear in the number of bits.
- Since the maximum for $d$ is $2^n$, $v$ has $\log(2^n) = n$ bits.

The triple (pair) of prefixes $u$ tells us how the depths are related.

- Connect the paths for $u$ with the appropriate checks at the end.
Sources I


Stéphane Demri, Diego Figueira, and M. Praveen. Reasoning about data repetitions with counter systems.
Sources II


Richard Lipton.
The reachability problem is exponential-space hard.
*Yale University, Department of Computer Science, Report, 62, 1976.*