10.5 Robustness for Parameterized Programs

Motivation: Libraries cannot make assumptions on the number of threads executing their functions.

Goal: Solve robustness for parameterized programs.

Definition (Parameterized programs):

- Syntactically, a parameterized program $P = \{t_1, \ldots, t_k\}$ consists of a finite number of threads (like a normal parallel program).
- Semantically, a parameterized program represents a family of parallel programs, so-called instances: for every vector $I = (n_1, \ldots, n_k) \in \mathbb{N}^k$, instance $P(I)$ has n_i copies of thread t_i.

The parameterized robustness problem is as follows:

Given: A parameterized program P.

Problem: Does $Tr_{TSO}(P(I)) = Tr_{SC}(P(I))$ hold for all instances $P(I)$ of P?

Approach:

- With instrumentation, instance $P(I)$ is not robust iff there is an attack A so that $P(I)_A$ reaches a goal configuration.
- Impossible to instrument every instance $P(I)$, infinitely many. Instead, instrument P directly, turning P into P_A which is now a parameterized program.

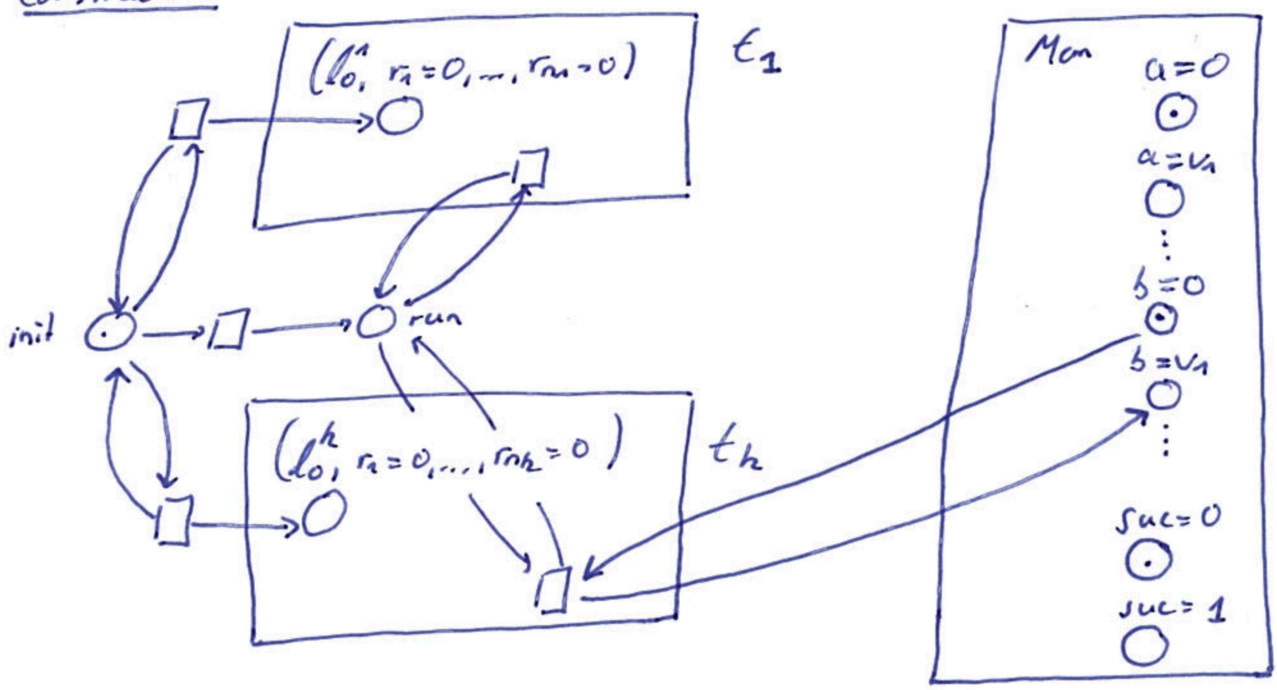
Careful:
↳ Only one copy of t_A should act as attacker.
The remaining copies of t_A have to behave like helpers.
↳ Hence, t_A has to be instrumented both, as attacker and as helper.
↳ To make sure a single thread becomes the attacker, introduce a flag that is set atomically from 0 to 1 (by the thread acting as attacker).
Atomic instructions like test-and-set can be added to our programming model without harm for the theory.

- Apart from this change $P(I)_A$ and $P_A(I)$ coincide.

Theorem:
A parameterized program P is not robust
iff there is an attack A so that
an instance $P_A(I)$ of the parameterized program P_A
reaches a goal configuration under SC.

From parameterized programs over finite domains to Petri nets:
Consider parameterized programs over finite domains.
Goal: Show that (SC) reachability of a goal configuration in an instance of P_A
can be formulated as a coverability problem for Petri nets.
Idea: Threads never use their identifiers,
hence there is no need to track their identities.
Instead, we count how many instances of each thread are in a certain (local) configuration.
The technique is known as counter abstraction

Construction:



Theorem:

Robustness for parameterized programs over finite domains is decidable in 2EXPSPACE and EXPSPACE-hard, already in the Boolean case.

Proof:

Coverability for Petri nets is EXPSPACE-complete.

↳ We reduce parameterized robustness to coverability in an exponentially large Petri net.
This gives a 2EXPSPACE upper bound.

↳ For the lower bound, we reduce coverability to SC-reachability for parameterized programs (with atomic test-and-set).
Then hardness of robustness follows like in the non-parameterized case.

□