TECHNISCHE UNIVERSITÄT DRESDEN

Concurrency Theory

Lecture 7: Checking Bisimilarity is P-Complete*

Dr. Stephan Mennicke

Institute for Theoretical Computer Science
Knowledge-Based Systems Group

May 26, 2025

International Center
for Computational Logic

Recall and Conclude

Theorem 33 CCS is Turing-complete. For every Minsky machine $\mathcal{M}$ there is a process $P(\mathcal{M})$ with a special constant L_H representing the halting line of $\mathcal{M}$ such that $\mathcal{M}$ terminates if and only if $P(\mathcal{M}) \xrightarrow{\tau}{}^{\star} \xrightarrow{\checkmark}$.

Everything *interesting* about CCS is undecidable.

The Bisimilarity Problem

Input Processes $p, q \in \mathsf{Pr}$.

Output Yes if and only if $p \simeq q$.

Recall, everything *interesting* about CCS is undecidable. Thus,

Theorem 34 $\simeq$ is undecidable for CCS processes.

What makes $\simeq$ undecidable?

Our simulation of Minsky machines made heavy use of

1. process constants (to mimic recursion)
2. synchronization (to enforce proper moves)

During the next two lectures, we restrict CCS in both respects to explore the boundary of undecidability of $\simeq$.

First, consider the finite fragment of CCS

$$\mathsf{CCS}_{fin} = \mathsf{CCS}(\mathsf{Act}, \emptyset, \emptyset)$$

Recall: Bisimilarity in P for Finite LTSs

$\simeq_\omega := \bigcap_{i \geq 0} \simeq_i$

1. set $\simeq_0 = \mathcal{U}$
2. $p \simeq_{n+1} q$ for $n \geq 0$ if for all $a \in \mathsf{Act}$:
   a. for all p' with $p \xrightarrow{a} p'$, there is a q' with $q \xrightarrow{a} q'$ and $p' \simeq_n q'$;
   b. for all q' with $q \xrightarrow{a} q'$, there is a p' with $p \xrightarrow{a} p'$ and $p' \simeq_n q'$.
3. Finite LTSs are image-finite. **recall**
4. How hard is it to compute $\simeq$ on finite LTSs $(\mathsf{Pr}, \mathsf{Act}, \longrightarrow)$? i.e., $\simeq_\omega$
   - compute $\simeq_0 = \mathcal{U}$ $\mathcal{O}(|\mathsf{Pr}|^2)$
   - iteratively remove all pairs from $\simeq_i$ contradicting bisimulations $\not\rightsquigarrow \simeq_{i+1}$ $\mathcal{O}(|\mathsf{Pr}|^3)$
   - stop when nothing changes after at most $|\mathsf{Pr}|^2$ removals

Recall: Bisimilarity in P for Finite LTSs

Lemma 35 For all processes $P \in \mathsf{CCS}_{fin}$, $G(P)$ is finite and finitely branching.

Proof: By induction on the structure of P.

Base $P = \mathbf{0}$ has a single node (i.e., process) and no transitions.

Step Let $Q_1, Q_2 \in \mathsf{CCS}_{fin}$ with finite(ly branching) $G(Q_i)$ $(i = 1, 2)$.

$\boldsymbol{P = \mu.Q_1}$ *easy*

$\boldsymbol{P = \nu a\, Q_1}$ *easy*

$\boldsymbol{P = Q_1 + Q_2}$ *easy*

$\boldsymbol{P = Q_1 \mid Q_2}$ recall, every process can be turned into *head normal form*; for CCS_{fin} processes, repeated application eventually terminates with a process Q without any parallel composition operator.

∎

Main Outcome of Today

Theorem 36 $\simeq$ is P-complete for CCS_{fin} processes.

Proof: The procedure we sketched is polynomial. It remains to be shown that $\simeq$ can be used to solve any problem in P, i.e., it is P-hard. ∎

Roadmap

1. Detour: game characterization of $\simeq$
2. *Monotone Circuit Value Problem* (MCVP) is P-hard
3. Reduction from MCVP completes the proof of Theorem 36

Along the lines, we learn a proof technique that is special to bisimilarity proofs, namely *defender's forcing*.

Detour: The Bisimulation Game

Let $\mathcal{T} = (\mathsf{Pr}, \mathsf{Act}, \longrightarrow)$ be an LTS. We call $\mathcal{B} := \mathsf{Pr} \times \mathsf{Pr}$ the game board of the *bisimulation game*, being a 2-player game between R (the *refuter*) and V (the *verifier*), played on a board position $(P, Q) \in \mathcal{B}$.

1. R chooses a transition $P \xrightarrow{\mu} P'$ or $Q \xrightarrow{\mu} Q'$;
2. V has to find a matching transition, either $Q \xrightarrow{\mu} Q'$ or $P \xrightarrow{\mu} P'$.

A *play for* (P_0, Q_0) is a finite or infinite sequence of pairs

$$(P_0, Q_0), (P_1, Q_1), ..., (P_i, Q_i), ...$$

In a play R tries to show that P_0 and Q_0 are not equal (i.e., bisimilar) while V tries to show the opposite.

If, at some point, V is unable to answer, R *wins*. Otherwise, if R cannot enforce a *win*, V *wins*.

Detour: (Winning) Strategies

A strategy for R specifies, for all possible plays

$$(P_0, Q_0), (P_1, Q_1), ..., (P_i, Q_i)$$

which transition to choose as the next challenge.

A strategy for V specifies, for all possible plays

$$(P_0, Q_0), (P_1, Q_1), ..., (P_i, Q_i)$$

and challenges (by R), which transition to choose as the next answer.

A strategy (for R or V) is called a *winning strategy* if it leads to a win in all possible plays.

Detour: (Winning) Strategies

Theorem 37 $P \simeq Q$ if and only if V has a winning strategy for (P, Q).

Theorem 38 $P \not\simeq Q$ if and only if R has a winning strategy for (P, Q).

Intermediate Step

Theorem 36 $\simeq$ is P-complete for CCS_{fin} processes.

Roadmap

1. Detour: game characterization of $\simeq$
2. *Monotone Circuit Value Problem* (MCVP) is P-hard
3. Reduction from MCVP completes the proof of Theorem 36

Along the lines, we learn a proof technique that is special to bisimilarity proofs, namely *defender's forcing*.

Computing with Circuits

Slides 10 to 25 from lecture 19 of Complexity Theory, WS 2024/2025 from January 6, 2025.

(Monotone) Circuit Value Problem = (M)CVP

Input A (monotone) Boolean circuit C with n inputs and one output, as well as an input string $i \in \{0,1\}^n$.

Problem Does $C(i)$ evaluate to 1?

From Circuits to Montone Circuits?

Discussion

(Monotone) Circuit Value Problem = (M)CVP

Theorem 39 MCVP is P-hard.

Proof: Reuse the DTM simulation of Theorem 19.12, borrowed from the 19th lecture of complexity theory, **taught next winter again**. ∎

Intermediate Step

Theorem 36 $\simeq$ is P-complete for CCS_{fin} processes.

Roadmap

1. Detour: game characterization of $\simeq$
2. *Monotone Circuit Value Problem* (MCVP) is P-hard
3. Reduction from MCVP completes the proof of Theorem 36
   - For every monotone circuit C with output node o and input string i, construct a *finite* LTS with two states P_o and Q_o
   - in such a way that $C(i) = 1$ if and only $P_o \simeq Q_o$.
   - In fact, the construction will also include many more equivalence notions, such as the *simulation preorder*.

Along the lines, we learn a proof technique that is special to bisimilarity proofs, namely *defender's forcing*.

Reduction from MCVP C and $i \in \{0,1\}^n$

The LTS we consider is the smallest LTS $\mathcal{T}(C,i) = (\mathcal{Q}, \{\ell, r, a, 0\}, \longrightarrow)$ such that

1. $P_{\text{end}} \in \mathcal{Q}$;
2. $P_v, Q_v \in \mathcal{Q}$ for every node v of C, and additionally,
3. $P'_v, Q^\ell_v, Q^r_v \in \mathcal{Q}$ for every node v of C labeled with $\vee$.

The transition relation $\longrightarrow$ contains the following transitions:

1. $P_v \xrightarrow{\ell} P_{v_1}, P_v \xrightarrow{r} P_{v_2}, Q_v \xrightarrow{\ell} Q_{v_1}, Q_v \xrightarrow{r} Q_{v_2}$ for every node v of C with label $\wedge$;
2. $P_v \xrightarrow{a} P'_v, P_v \xrightarrow{a} Q^\ell_v, P_v \xrightarrow{a} Q^r_v$, and
   $P'_v \xrightarrow{\ell} P_{v_1}, P'_v \xrightarrow{r} P_{v_2}$, and
   $Q_v \xrightarrow{a} Q^\ell_v, Q_v \xrightarrow{a} Q^r_v$, and
   $Q^\ell_v \xrightarrow{\ell} Q_{v_1}, Q^\ell_v \xrightarrow{r} P_{v_2}, Q^r_v \xrightarrow{r} Q_{v_2}, Q^r_v \xrightarrow{\ell} P_{v_1}$ for every node v of C with label $\vee$;
3. $P_v \xrightarrow{0} P_{\text{end}}$ for every input node v of C with assigned value 0.

The construction of $\mathcal{T}(C,i)$ can clearly be computed in log-*space*.

Reduction from MCVP C and $i \in \{0,1\}^n$

Theorem 36 $\simeq$ is P-complete for CCS_{fin} processes.

Proof: Membership in P given. Hardness:

- If $C(i) = 1$, then $P_o \simeq Q_o$.
- If $C(i) = 0$, then $P_o \not\simeq Q_o$.

Our construction works, even if player R stays on the side of all P-processes.

1. Discuss $\wedge$-nodes: who decides the next pair in the play?
2. Discuss $\vee$-nodes: who decides the next pair in the play?

What makes $\simeq$ undecidable?

Our simulation of Minsky machines made heavy use of

1. process constants (to mimic recursion)
2. synchronization (to enforce proper moves)

Next lecture, we explore the boundary of undecidability of $\simeq$.

1. We allow recursion, but disallow synchronization.
2. We explore the realm of Petri nets a bit.
3. A lot of useful lemmas are on the horizon.