Distributed Synthesis for LTL Fragments

Krishnendu Chatterjee, Thomas A. Henzinger, Jan Otop, Andreas Pavlogiannis

July 18, 2013
Synchronous architecture $\mathcal{A} = (\mathcal{P}, p_e, V, E)$

- $\mathcal{P}$ is a set of $n + 1$ processes.
- $p_e \in \mathcal{P}$ is the environment.
- $V$ is a set of binary variables.
- $E : \mathcal{P} \times \mathcal{P} \to 2^V$ defines the communication.
- For $p \in \mathcal{P}$ denote input variables with $I(p)$, output variables with $O(p)$. 
Process $p$ behaves according to local strategy $\sigma_p : (2^{I(p)})^* \rightarrow 2^{O(p)}$.

Can be viewed as the labeling of an infinite $2^{I(p)}$-tree, $T_{\sigma_p}$.

The collective strategy $\sigma : (2^{O(p_e)})^* \rightarrow 2^{V \setminus O(p_e)}$ determines the distributed behavior of the system.

Can be viewed as the labeling of an infinite $2^{O(p_e)}$-tree, $T_\sigma$. 
Realizability

- Every infinite path $\pi = (a_1, a_2 \ldots)$ of $T_\sigma$ defines a *computation* $\ell_\sigma(\pi) = (c_1, c_2 \ldots)$.
  - $\ell_\sigma(\pi)[i]$ is the set of variables being True at that node.
- Acceptable computations are specified by LTL specifications.

Realizability

Given an architecture $\mathcal{A}$ and an LTL specification $\phi$, decide whether there exist local strategies $\sigma_p$ for all processes $p$, such that for every path $\pi$ in the corresponding collective strategy tree $T_\sigma$, it holds $\ell_\sigma(\pi) \models \phi$.

- If so, synthesize them.
Distributed realizability was shown to be undecidable for the following architecture.

Reduction from the halting problem.

For any Turing machine $M$, construct $\phi_M$ which requires that $p_1$, $p_2$ output a legal sequence of configurations of $M$, and $M$ halts.

1. When $p_i$ receives a start signal, it outputs a sequence of legal configurations of $M$.
2. Initially $p_i$ outputs the first two configurations of $M$.
3. If $p_1$, $p_2$ output $C_1C_1'$ and $C_2C_2'$ and $C_1 \vdash C_2$, then $C_1' \vdash C_2'$.
For which classes of architectures is realizability decidable?

Complete characterization base on the *information fork* criterion.

Processes $p_1$, $p_2$ form an information fork in architecture $A$ if there exist paths $p_e \leadsto p_i$ in $A$ such that do not traverse edges in $I(p_i)$.

Every architecture either:
- Has an information fork (undecidable).
- Can be reduced to a pipeline (decidable).
For propositional formulae $P$ and $Q$, we consider $\phi \in \text{LTL}^{\Diamond}$ of the form

$$
\theta = P \mathbin{|} \mathcal{X}P \\
\psi = \theta_1 \land \theta_2 \mathbin{|} \theta_1 \lor \theta_2 \mathbin{|} \neg \theta \\
\phi = Q \rightarrow \Diamond \psi
$$

Theorem

The realizability of specifications from $\text{LTL}^{\Diamond}$ in some architecture $\mathcal{A}$ is decidable iff $\mathcal{A}$ does not have information fork.

K. Chatterjee, T. A. Henzinger, J. Otop, A. Pavlogiannis
Distributed Synthesis for LTL Fragments
Fix a Turing machine $M$, with tape alphabet $\{0, 1, \sqcup\}$ and set of states $Q$. Let $\Sigma = \{0, 1, \sqcup, \bot, \#\} \cup Q$.

Configurations of $M$ are words over $\Sigma \setminus \{\bot\}$ and start with $\#$.

Projection $\pi_\bot : \Sigma^* \to (\Sigma \setminus \{\bot\})^*$ omits the $\bot$ symbols.

A scattered configuration $C$ is a word over $\Sigma$ such that $\pi_\bot(C)$ is a configuration of $M$. Denote with $\bot(C) = \{i : C[i] = \bot\}$.

A scattered preconfiguration is a word over $\Sigma$ which is a prefix of some scattered configuration.

We write $C_1 \parallel C_2$ if $|\bot(C_1) \bigtriangleup \bot(C_2)| \leq 1$.

### $C_1 \vdash C_2$

For scattered preconfigurations $C_1$ and $C_2$ we write $C_1 \vdash C_2$ if

1. $\pi_\bot(C_1) \vdash \pi_\bot(C_2)$, or

2. $C_1$ and $C_2$ are infinite and every finite prefix of $\pi_\bot(C_i)$ can be extended to $C'_i$ such that $C'_1 \vdash C'_2$. 
We consider the following architecture

\[ p_1 \rightarrow p_e \rightarrow p_2 \]

- \( p_e \) sends \textit{next} and \textit{stall} signals.
- Processes output infinite words over \( \Sigma \).
- Undecidability obtained through reduction from the halting problem of \( M \).
- We first describe a safety property \( \varphi \).
Proof Idea (Cont.)

\[ \varphi = \mathcal{L} \rightarrow \bigwedge_{0 \leq i \leq 4} \text{Cond}_i \]

**\mathcal{L}:** for every process, every *stall* input signal is followed by a *next* signal.

**Cond\(_0\):** each process outputs \( \bot \) when its input is *stall*, otherwise it outputs a letter from \( \Sigma \setminus \{\bot\} \),

**Cond\(_1\):** each process produces a sequence of scattered preconfigurations,

**Cond\(_2\):** initially each process produces two scattered configurations of \( M \), whose projections are the first two valid configurations of \( M \),

**Cond\(_3\):** if starting from some position, \( p_1 \) outputs consecutively \( C_1, C_2 \) and \( p_2 \) outputs consecutively \( C'_1, C'_2 \), then \( C_1 \vdash C'_1 \) implies \( C_2 \vdash C'_2 \) or \( C'_2 \parallel C_2 \),

**Cond\(_4\):** if \( D, D' \) are outputs of \( p_1, p_2 \) up to some positions such that \( D \parallel D' \) and \( |\pi_\bot(D)| = |\pi_\bot(D')| \), then \( \pi_\bot(D) = \pi_\bot(D') \).
Proof Idea (Cont)

- $\varphi$ can be expressed by a safety automaton $A_{safe}$.

**Lemma**

Strategies realizing $A_{safe}$ output a valid computation of $M$.

- $Cond_2$ requires the first two configurations.
- If $C_1$ and $C'_1$ are the $i$ and $i + 1$ configurations of $M$, then $p_e$ can synchronize them as outputs of $p_1$ and $p_2$.
- Needs to send at most $i$ stalls to $p_1$, without violating $L$.
- If no more stalls follow, by $Cond_3$, $C'_1 \vdash C_1$ implies $C'_2 \vdash C_2$.
- Due to $Cond_4$ we can show that for any execution under $L$, $\pi_\perp(C'_1)$ and $\pi_\perp(C'_2)$ are the $i + 1$ and $i + 2$ configurations of $M$. 
\( \phi = Q \rightarrow \Diamond (\psi_1 \lor \psi_2) \)

\( Q \) The first state of \( A_{\text{safe}} \) according to the output variables \( \{q_1, \ldots, q_m\} \) corresponds to the first step of the computation.

\( \psi_1 \) \( p_e \) cheats in simulating \( A_{\text{safe}} \).

\( \psi_2 \): The current state of \( A_{\text{safe}} \) is not rejecting, and \( p_1 \) or \( p_2 \) output a halting state of \( M \).

\[ q_1, \ldots, q_m \]

\( \phi \) is realizable iff \( M \) halts.
For propositional formulae $P$ and $Q$, we consider $\phi \in \text{LTL} \Box$ of the form

$$
\psi = P \mid \psi_1 \land \psi_2 \mid \psi_1 \lor \psi_2 \mid \neg \psi \mid X\psi
$$

$$
\phi = Q \land \Box \psi
$$

- Consider star architectures with $p_e$ the central process.
- We distinguish between overlapping inputs (inter process communication), and disjoint inputs.
\( \phi = Q \land \Box(\psi_1 \land \psi_2 \land \psi_3) \)

**Q**: The first state of \( A_{\text{safe}} \) according to the output variables \( \{q_1, \ldots, q_m\} \) corresponds to the first step of the computation.

**\( \psi_1 \)**: \( p_3 \) simulates \( A_{\text{safe}} \) faithfully in \( \{q_1, \ldots, q_m\} \).

**\( \psi_2 \)**: \( p_1 \) and \( p_2 \) do not output a halting state of \( M \).

**\( \psi_3 \)**: \( A_{\text{safe}} \) does not reach a rejecting state.
Let $k$ be the nesting depth of $\chi$ operators in $\psi$.

**Lemma**

A formula $\phi = Q \land \Box \psi$ is realizable iff it is realizable by bounded strategies of depth $k + 2^{k|V|}$.

- Assume $\phi$ is realizable by local strategies $\sigma_i$.
- If some good computation $\ell_{\sigma}(\pi)$ repeats a $k$-segment then another computation $\ell_{\tau}(\pi)$ that loops between the two segments is also good.
The type of a local node is the unique history of inputs and outputs $k$ steps back.

- There exist at most $2^{k|V|}$ unique types of nodes.
- The type of a level is the set of types of all local nodes in that level.
- There exist at most $2^{2^k|V|}$ unique types of levels.

Define folding functions $f_i$ that folds levels with the same types, wrt the types of the nodes.