Boolean Synthesis

Given: Boolean formula \( F(\vec{x}, \vec{y}) \) representing a relation over input variables \( \vec{x} = \{x_1, \ldots, x_m\} \) and output variables \( \vec{y} = \{y_1, \ldots, y_n\} \).

Obtain: Boolean function \( g : \{0, 1\}^m \rightarrow \{0, 1\}^n \) such that, for all \( \vec{x} \), \[ F(\vec{x}, g(\vec{x})) \Leftrightarrow \exists \vec{y}. F(\vec{x}, \vec{y}) \]

- \( F \) is called the specification.
- \( g \) is called the implementation.

Example: The two’s complement of a two-bit integer \( x_0x_1y_0 \) is a two-bit integer \( y_1y_0 \) such that \( x_0x_1 + y_1y_0 = 0 \). We can synthesize a function that computes the two’s complement as follows:

\[
F(x_0, x_1, y_0, y_1) = \neg(x_0 \oplus y_0) \land \neg(x_1 \oplus y_1 \land (x_0 \land y_0))
\]

\[
g(x_0, x_1) = \begin{cases} y_0 := x_0 \\ y_1 := x_1 \oplus x_0 \end{cases}
\]

Despite extensive research on the subject, Boolean synthesis remains a challenging NP-hard problem.

A standard strategy for handling hard problems is decomposing them into smaller problems. Our goal is to apply this concept to Boolean synthesis.

Decomposition using Factored Formulas

One way to decompose Boolean synthesis is to use factored formulas [2, 3]:

\[ F(\vec{x}, \vec{y}, \vec{z}, \vec{y}) = F_1(\vec{x}, y_2, y_1) \land F_2(\vec{x}, y_1, y_2, y_0) \land F_3(\vec{x}, y_1) \]

Pros:
- Easy to perform decomposition.
- Specifications are often already given as a conjunction of constraints.
- Each factor uses only a subset of the variables.

Cons:
- Dependencies between factors.
- Highly non-trivial to combine implementations of \( F_1, \ldots, F_k \) into an implementation of \( F \) [2].

This form of decomposition has been shown to significantly improve synthesis algorithms [2, 3]. However, dealing with the dependences between factors prevents us from taking full advantage of the decomposition [3].

Towards Sequential Decomposition

Given: Boolean formula \( F(\vec{x}, \vec{y}) \) representing a relation over input variables \( \vec{x} = \{x_1, \ldots, x_m\} \) and output variables \( \vec{y} = \{y_1, \ldots, y_n\} \).

Obtain: Formulas \( F_1(\vec{x}, \vec{z}) \) and \( F_2(\vec{z}, \vec{y}) \) for intermediate variables \( \vec{z} = \{z_1, \ldots, z_k\} \) such that, if \( g_1 \) implements \( F_1 \) and \( g_2 \) implements \( F_2 \), then \( g_2 \circ g_1 \) implements \( F \).

Sequential Decomposition

References

