Symbolic Synthesis of Distributed Systems with Petri Games

Manuel Gieseking

Motivation

Distributed Synthesis
- **Approach**: Given a formal specification, automatically derive correct-by-construction implementation (if existent).
- **Scenario**: Distributed system with multiple concurrent processes acting independently.
- **Goal**: Derive local controller for each process, collectively fulfilling the global specification.

Petri games
- Extension of Petri nets to *multi-player games*.
- Each token is a player.
- Players have causal memory (possibly obtained through synchronization).
- Solving takes only single-exponential time.

Related Work: Other Frameworks for Distributed Synthesis
- **Deciding the Distributed Synthesis Problem – Symbolically Solving Petri Games**
- **Evaluation**
- **Case Studies**
- **Future Work**

Deciding the Distributed Synthesis Problem – Symbolically Solving Petri Games

Example:
Synthesis of local controllers for a distributed alarm system (AS):
- A **burglar** (environment player) intrudes one of \( n \) secured locations (here \( n = 2 \)).
- The corresponding **local alarm system** (system player) should detect the burglary and communicate the intruded location to the other alarm systems (system players).
- **Goal**: Each alarm system should correctly indicate the burglar’s intrusion (no false alarm or false report).

Evaluation

Petri game strategy for system players.

Example:
Petri game strategy for system players.

Reduction

Symbolic game solving

Part of a 2-player game over finite graph.

Tool Support: ADAM

"Adam is named in honor of Carl Adam Petri (1926–2010)."

The tool ADAM (Analyzer of Distributed Asynchronous Models)
- automatically synthesizes local controllers from a given Petri game,
- uses symbolic game solving algorithm with BDDs,
- handles case studies with up to 33 system processes (30 minutes time out),
- achieved artifact evaluation badge from CAV (Computer Aided Verification).

Available at [http://www.uni-oldenburg.de/csd/adam/](http://www.uni-oldenburg.de/csd/adam/)

Related Work: Other Frameworks for Distributed Synthesis
- **Zielonka automata**: asynchronous concurrency with shared actions and causal memory. General: decidability open, tree architectures: nonelementary.

Case Studies

- **CM**: \( n \) concurrent machines process \( k \) orders, each order by one machine. The environment decides which machines are functioning.
- **SR**: Self-reconfiguration of \( n \) robots on which the environment destroys up to \( k \) tools.
- **JP**: Processing of a job by a subset of \( n \) processors selected by the environment.
- **DW**: Workflow of a document among \( n \) clerks starting at a clerk chosen by the environment (DWS a simpler variant).

Future Work

- New reduction to a 2-player game over finite graphs (by preserving the complexity result).
- New winning conditions (reachability, Büchi, parity, global objectives).
- Extend level of informedness (add forgetful places, indistinguishable transitions (partial observation)).

Manuel Gieseking
gieseking@informatik.uni-oldenburg.de