Symbolic Synthesis of Finite-State Controllers for Request-Response Specifications

Nico Wallmeier, Patrick Hütten, Wolfgang Thomas
Chair of Computer Science 7
RWTH Aachen, Germany

RWTH RHEINISCH-WESTFÄLISCHE TECHNISCHE HOCHSCHULE AACHEN

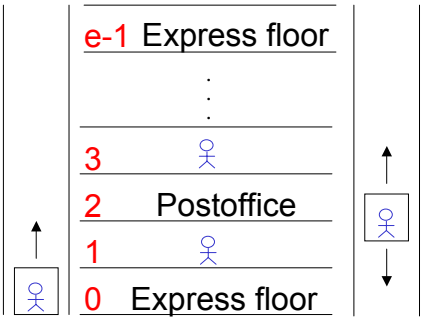
i7

Structure

1. Motivation
2. Infinite two person games
3. Request-Response games
4. Symbolic method
5. Case study
6. Conclusion

Goal

Goal is to compute a controller program for such examples:



- Two players (act in alternating moves):
  - Lift controller
  - Environment

Specification

Requirements to the lift controller:

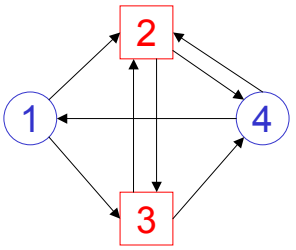
- After a floor is requested, a lift will eventually come to this floor.
- Top and ground floor are served directly.
- Second floor (post office): A lift stopping there waits one extra move before proceeding.
- Both lifts never stop at the same time on the second floor.
- No lift skips a requested floor on its way.
- A lift moves to an unrequested floor only if there is no open request.

Assumptions on the environment's behaviour:

- At most one person enters a lift at a time.
- Not all floors are requested simultanously.

Infinite Two Person Games

- System model: 2 player
  - Controller (player 0)
  - Environment (player 1)
- **Game graph** (arena)
  - Set of states $Q = Q_0 \cup Q_1$
  - Transitions $E \subseteq Q \times Q$
    (every state must have a successor)



- **Play** ρ is an infinite sequence of states $\rho=\rho(0)\rho(1)\rho(2)...$ with $(\rho(i),\rho(i+1))\in E$
- **Winning condition** for player 0

Infinite Two Person Games - 2

- **Winning strategy** of player i from state v: a function which tells player i for each play prefix from v how to proceed.
- **Winning region** of player i: set of all states from which player i has a winning strategy.

➢ Computation of winning regions and winning strategies leads to automatic synthesis of control programs.

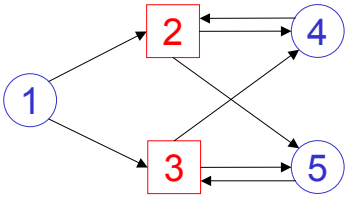
Method for Solving Example

1. Capture safety conditions by restricting the game graph
2. Rest of winning conditions is conjunction of request-response conditions: Reduce to Büchi condition
3. Solve game for Büchi condition

Request-Response Games

- A **Request-Response game** has the following winning condition:
  - Sets $P_i, R_i \subseteq Q$ for $1 \leq i \leq r$
  - (P_i, R_i) is called a **RR pair** or **RR condition**
  - Winning condition given by
    - $\bigwedge_{i=1}^{r}$ if play π reaches P_i at time j, then also R_i at a time $j' \geq j$
    - "whenever a state in P_i is visited, then eventually a state in R_i is visited"
    - Or in LTL: $\bigwedge_{i=1}^{r} G(P_i \rightarrow F\, R_i)$
- Similar to Streett games

Example



- "whenever state 1 is visited, then eventually state 4 is visited"
- "whenever state 2 is visited, then eventually state 4 or 5 is visited"
- RR conditions:
  - $P_1=\{1\}$ $R_1=\{4\}$
  - $P_2=\{2\}$ $R_2=\{4,5\}$
- Winning region
  - Player 0: {2,3,4,5}
  - Player 1: {1}

Reduction to Büchi Games

- **Büchi condition**: Some state of a set $F \subseteq Q$ is visited infinitely often
- Algorithmic solution for Büchi games is well known (fixpoint computation)
- Büchi games can be solved in polynomial time

Reduction to Büchi Games - 2

Theorem: RR games are reducible to Büchi games, involving a blow-up from n to $nr2^{r+1}$ states if r RR conditions are involved.

Idea: Keep in mind (enconding in an expanded game graph):

- Which requests are not yet fulfilled
- Which one should be fulfilled next (progressing cyclically, thereby visiting final states)

Reduction to Büchi Games - 3

Sketch of Proof: Expanded game graph $G'=(Q',E')$ and set F of final states:

- $Q' = Q \times 2^{\{1,\ldots,r\}} \times \{1,\ldots,r\} \times \{0,1\}$
- $((q,M,m,f),(q',M',m',f')) \in E' \Leftrightarrow$
  - $(q,q') \in E$
  - $M' = (M \cup \{ i \mid q' \in P_i\}) \setminus \{ i \mid q' \in R_i\}$
  - $m' = \begin{cases} m & \text{, if } m \in M' \\ (m \bmod r) + 1 & \text{, otherwise} \end{cases}$
  - $f' = \begin{cases} 0 & \text{, if } m = m' \\ 1 & \text{, otherwise} \end{cases}$
- $F = Q \times 2^{\{1,\ldots,r\}} \times \{1,\ldots,r\} \times \{1\}$

Reduction to Büchi Games - 4

Theorem: There is a family of RR games s.t.:

- The number of nodes is linear in r.
- The number of RR conditions is linear in r.
- Every strategy automaton of player 0 has at least $2^r \cdot r$ states.

➢ The exponential blow-up in r is inevitable.

Symbolic Method

- Try to overcome “state explosion problem”
- Already well known from Model Checking
- States and Transitions described by Boolean formulas over a set of Boolean variables $V=\{v_0,\ldots,v_n\}$
- Game graph $G=(V, \varphi_0, \varphi_1, \tau)$
  - V: Boolean variables
  - φ_i: Boolean formula for the states of player i
  - τ: Transition formula
- Data structure: OBDDs (Ordered Binary Decision Diagrams)

Symbolic Method - 2

- Reduction RR to Büchi game, expanded game graph $G' = (V', \varphi_0', \varphi_1', \tau')$ for r RR conditions:
  - V' with $(n + r + \lceil \log r \rceil + 1)$ variables
    - $v_0 \ldots v_{n-1}$ for states of V ($|V|=n$)
    - $v_n \ldots v_{n+r-1}$ to encode the requests
    - $v_{n+r} \ldots v_{n+r+\lceil \log r \rceil -1}$ to encode the next request which should be fulfilled (binary encoding)
    - $v_{n+r+\lceil \log r \rceil}$ final state flag
  - Final state formula $\lambda = v_{n+r+\lceil \log r \rceil}$
  - Formulas φ_0', φ_1' and τ' are skipped here

Case Study

Coding of the lift controller problem:

Request-Response game with

- $3 \cdot (e-2)$ RR pairs if e floors are involved
- $2 \cdot \lceil \log e \rceil + 3e + 2$ Boolean variables
  - $2 \cdot \lceil \log e \rceil$: Current positions of the lifts
  - $3e$: Signaling the requests from inside the lifts and the floors
  - 1: Determine the player
  - 1: Capture waiting condition (post office)

Case Study - 2

Floors	Size game graph	BDD creation	Solve game	Size Büchi game	Size winning regions player 0	Size winning regions player 1
3	25	18.34s	24.27s	1,200	24	1
4	673	27.73s	40.39s	516,864	672	1
5	12,913	40.86s	699.34s	119,006,208	0	12,913

Environment's winning strategy for 5 floors:

- Environment forces one lift to the second floor without any other request
- Environment requests two floors
  - One express floor
  - The other one on the way to it

Conclusion

- Request-Response games
  - Algorithmic solution
  - Applied this for synthesis of finite state-controllers
  - Main bottleneck: number of RR conditions (exponential blow-up)
  - First implementation
- Open:
  - Refine analysis of RR games
  - More general winning conditions