Symbolic synthesis of state based reactive programs

Diploma Thesis

Nico Wallmeier

18.03.03

Structure

1. Background
2. Infinite games over a finite game graph
3. Transformation to the symbolic state space
4. Applications

1. Background

Motivation

- Crash of the first Ariane-5 rocket (iX, September 1996)
- Computation error of the Intel Pentium processor

$\Rightarrow$Verification is necessary

- Testing & Simulation
  - Does not supply any correctness guarantee
  - Sometimes only limited applicability

$\Rightarrow$ Computer aided techniques in formal verification

Model Checking

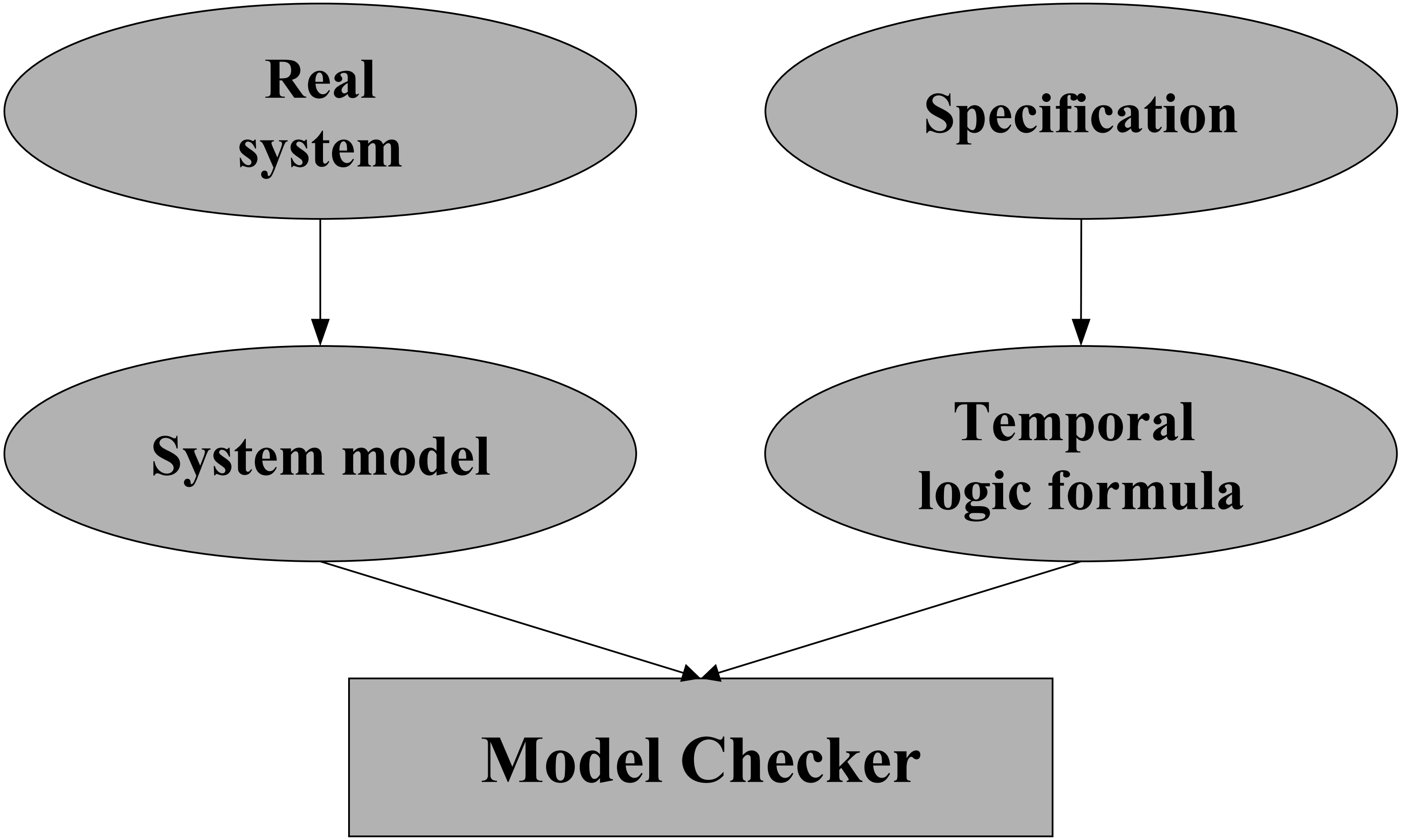
- Clarke, Emerson et al.:

  Model checking is an automatic technique for verifying correctness properties of safety-critical reactive systems.

- System is tested against a specification
- If an error occurs an error scenario will be generated

Model Checking - 2

- Procedure



Model Checking - 3

- Success in practical applications with two ideas (symbolic Model Checking):
  - Specification logic CTL (polynomial time) by Clarke and Emerson at the beginning of the 1980s
  - Symbolic method to overcome the "state explosion problem" – presentation of the states is done via BDDs (Binary Decision Diagrams) (Lee, Akers, Moret and Bryant)

Infinite two person games

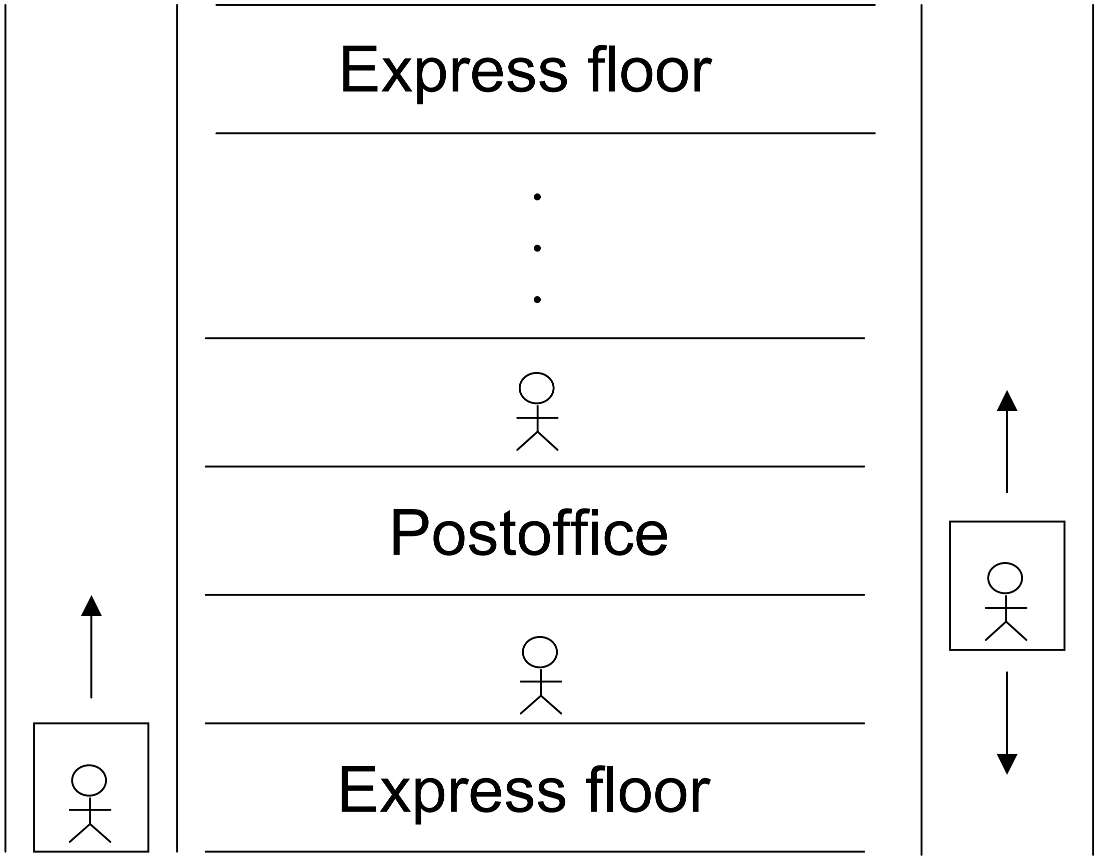
- Better system model: 2 agents
  - Controller (agent 0)
  - Environment (agent 1)
- Specification by
  - Game graph
  - Winning condition for player 0
- Play: Infinite path in the game graph

Infinite two person games - 2

- Classical theory of solving games
  - 1969 Büchi, Landweber
  - 1993 McNaughton
  - Currently: EU-project GAMES (Aachen, Bordeaux, ..., Warsaw)
- Goal of this work:
  - Transformation to the symbolic method
  - Implementation of these algorithms

Goal

- Goal is to solve such examples:



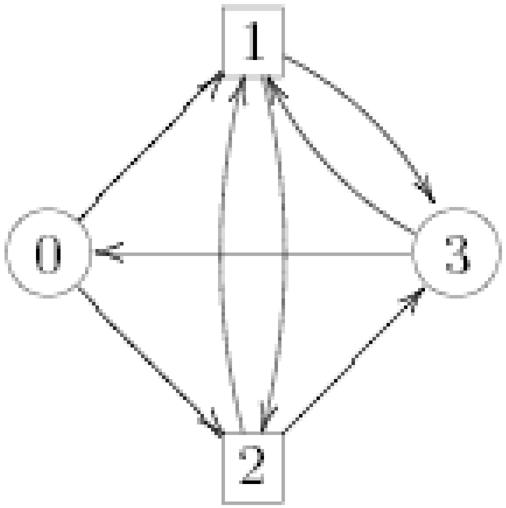
Specification

- Two lifts in a building with e floors should satisfy:
  - All requested floors will be served
  - The highest and the ground floor are served directly
  - No lift drives past a requested floor on his way
  - At most one person gets in a lift at a time
  - At least three floors are not requested
  - In the second floor is the post office. A lift needs one turn of the controller to wait there for exchanging the mail.
  - Both lifts are not at the same time in the second floor.

2. Infinite games over a finite game graph

Game graph

- *Game graph* G is defined by
  - Set of states $Q = Q_0 \dot{\cup} Q_1$
  - Transitions $E \subseteq Q \times Q$
    (every state must have a successor)



- *Play* ρ is a infinite sequence of states $\rho = \rho(0)\rho(1)\rho(2)\ldots$ with $(\rho(i),\rho(i+1)) \in E$
- $Oc(\rho) = \{ q \mid \exists i \; \rho(i) = q \}$ – occurrence set
- $In(\rho) = \{ q \mid \exists^{\omega} i \; \rho(i) = q \}$ – infinity set

Overview winning conditions

Name	Requirement	Winning condition
Reachability	$F \subseteq Q$	$Oc(\rho) \cap F \neq \emptyset$
Safety	$F \subseteq Q$	$Oc(\rho) \subseteq F$
Weak parity	$c:Q \rightarrow \{0,...,k\}$	$\max(Oc(c(\rho)))$ is even
Staiger-Wagner	$\mathcal{F} \subseteq Pot(Q)$	$Oc(\rho) \in \mathcal{F}$
Request-Response	$P_i, Q_i \subseteq Q$ $(1 \leq i \leq r)$	$\bigwedge_{i=1}^{r} \forall j\ (\rho(j) \in P_i \Rightarrow \exists j' \geq j\ \rho(j') \in R_i)$ Temporal: $\bigwedge_{i=1}^{r} G(P_i \rightarrow F\ R_i)$
Büchi	$F \subseteq Q$	$In(\rho) \cap F \neq \emptyset$
Parity	$c:Q \rightarrow \{0,...,k\}$	$\max(In(c(\rho)))$ is even

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

Reachability winning condition

- Simplest winning condition: reachability of a set F
- player 0 wins the play $\rho \Leftrightarrow$ ρ reaches a state in the set F sometime
- Solution with "Attractor": Compute for $i=0,1,2,\ldots$ the nodes, from which player 0 can reach the set F in $\leq i$ moves

Attractor

- Definition
  - $Attr_0^i(F) = \{ q \in Q \mid$ player 0 can reach the set F from q in $\leq i$ moves$\}$
  - $Attr_0^0(F) = F$
  - $Attr_0^{i+1}(F) = Attr_0^i(F)$
    $\cup \{ q \in Q_0 \mid \exists (q,r) \in E$ with $r \in Attr_0^i(F) \}$
    $\cup \{ q \in Q_1 \mid \forall (q,r) \in E$ holds $r \in Attr_0^i(F) \}$
- Conclusions:
  - $Attr_0^i(F) \subseteq Attr_0^{i+1}(F)$
  - $Attr_0^m(F) = Attr_0^{m+1}(F)$ for a $m \leq |Q|$
    $\Rightarrow Attr_0(F) = Attr_0^m(F)$ for such a m

Use of attractor computation

- Solvable games by attractor computation
  - Reachability game
  - Safety game
  - Weak parity game
  - Büchi game

3. Transformation to the symbolic state space

Motivation

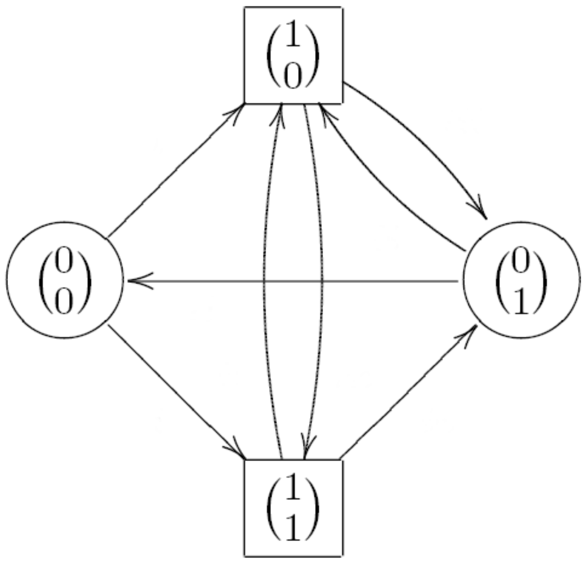
- Abstract state space:
  - „State Explosion Problem“
  - Analogous to Model Checking
  - Often no practical application possible

⇒In this work the symbolic method is introduced (as known from Model Checking)

Symbolic state space

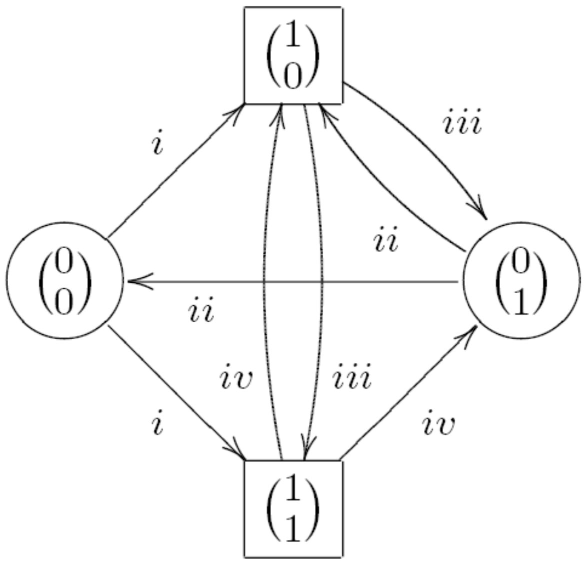
- Set of Boolean variables
- $V = \{v_0, \ldots, v_n\}$ as well as $V' = \{v'_0, \ldots, v'_n\}$
- Concrete state is an assignment of all variables of V
- 2^n states $\rightarrow$ n variables

Symbolic game graph



- Is defined by formulas for
  - Nodes of player 0
  - Nodes of player 1
  - Transitions
- Nodes of player 0
  - $\varphi_0 = \neg v_0$
- Nodes of player 1
  - $\varphi_1 = v_0$

Symbolic game graph - 2



- Transition formula τ

  i. $\neg v_0 \wedge \neg v_1 \wedge v_0'$

  ii. $\neg v_0 \wedge v_1 \wedge \neg v_1'$

  iii. $v_0 \wedge \neg v_1 \wedge v_1'$

  iv. $v_0 \wedge v_1 \wedge (v_0' \Leftrightarrow \neg v_1')$

Attractor

- Definition
  - $Attr_0^0(\lambda) = \lambda$
  - $Attr_0^{i+1}(\lambda) = Attr_0^i(\lambda) \vee$
    $(\varphi_0 \wedge (\tau \wedge Attr_0^i(\lambda)|_{V \to V'})|_V) \vee$
    $(\varphi_1 \wedge \neg(\tau \wedge \neg Attr_0^i(\lambda)|_{V \to V'})|_V)$
- Strategy
  - $Strat_0^0(\lambda) = false$
  - $Strat_0^{i+1}(\lambda) = Strat_0^i(\lambda) \vee$
    $(Attr_0^{i+1}(\lambda) \wedge \neg Attr_0^i(\lambda) \wedge \tau \wedge$
    $(\varphi_1 \vee (\varphi_0 \wedge Attr_0^i(\lambda)|_{V \to V'})))$

Achieved results

Game	Solution
Reachability	Attractor computation
Safety	Attractor computation
Weak parity	Attractor computation
Staiger-Wagner	Reduction to weak parity
Request-Response	Reduction to Büchi
Büchi	Attractor+ and Recur
Parity	McNaughton-algorithm

4. Applications

Input language

- x[2], x‘[2]
- Boolean Operations such as Or, And, XOr, XAnd, Not, ...
- Existential and universal quantifier for variable indices – e.g. Ei{i<3} x[i]
- Arithmetic for variable indices, e.g. x[i+3]
- External parameters

Case study

- Request-Response game with 3·(e-2) RR-pairs
- 5·e+3 variables for e floors
  - e variables: Position first lift
  - e variables: Position second lift
  - e variables: Requests on the floors
  - e variables: Requests in the first lift
  - e variables: Requests in the second lift
  - One variable to determine the player
  - Two variables for the post office

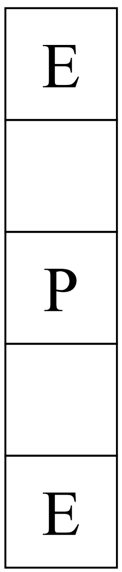
Case study - 2

Floors	Size game graph	BDD creation	Solve game	Size Büchi game	Size winning regions	
					player 0	player 1
3	25	40.69 s	30.38 s	1,200	24	1
4	673	53.77 s	73.09 s	516,864	672	1
5	12,913	172.29 s	191.10 m	119,006,208	0	12,913

Case study - 3

Winning strategy of the environment for five floors

Force one lift to second floor, let it wait one move with no other requests and look at second lift:

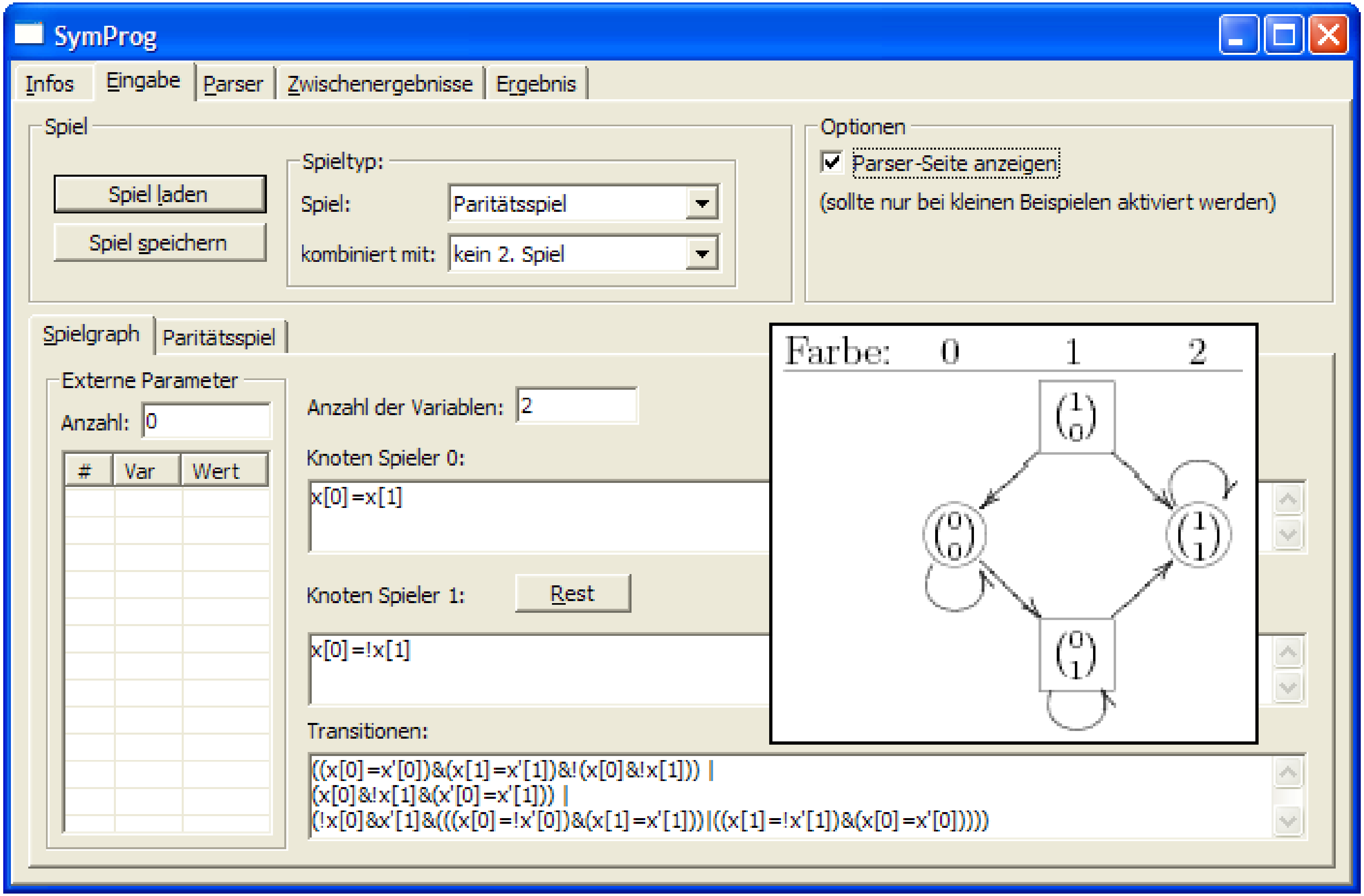


Pos. 2. Lift	Chosen Requests
Ground floor	1. floor + 4. floor
1. floor	Ground floor + 4. floor
3. floor	Ground floor + 4. floor
4. floor	Ground floor + 1. floor

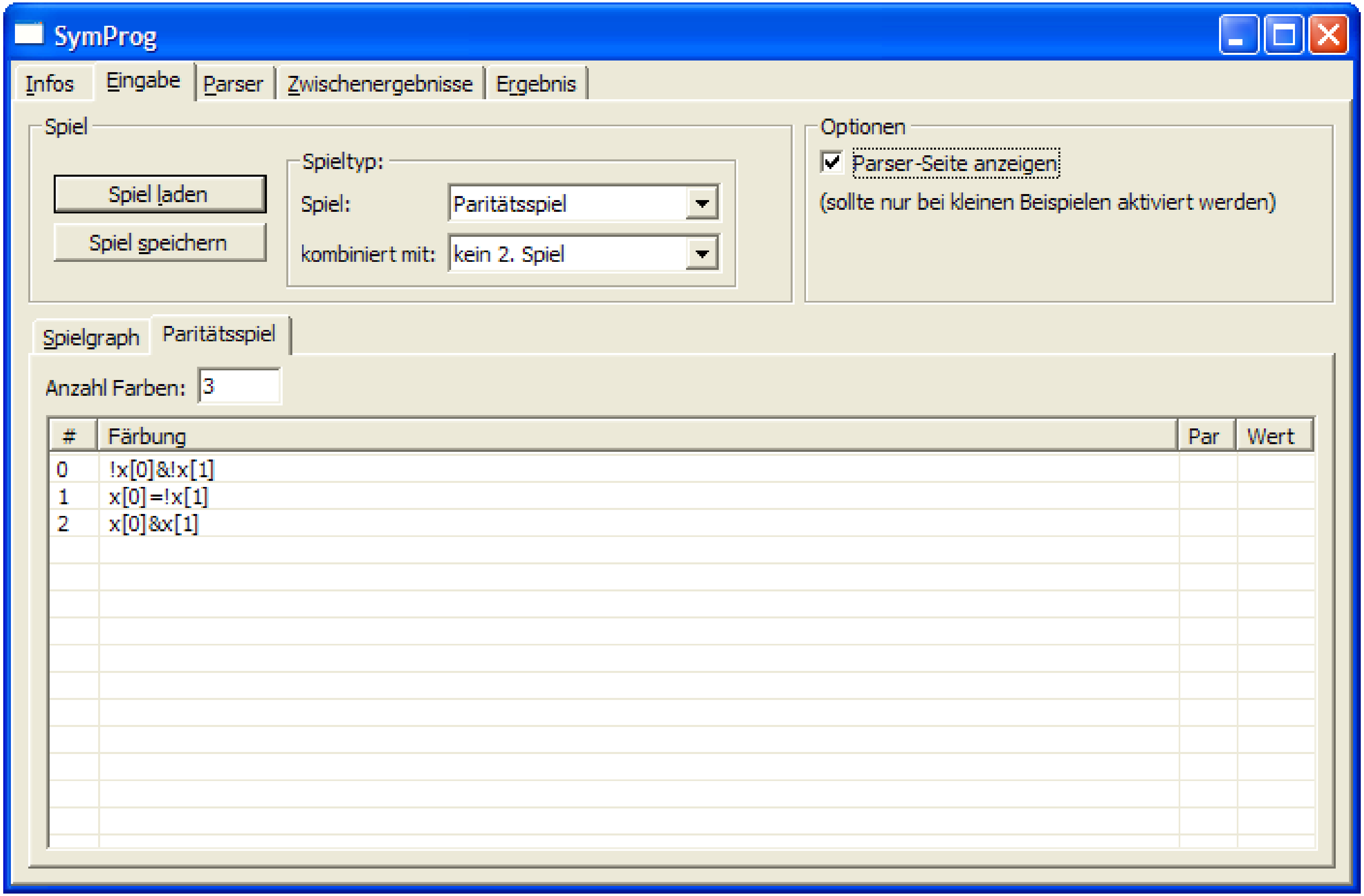
Further work

- Develop suitable restrictions for
  - Game graph specification
  - Winning conditions
- Hierarchical approach (SDL specification)
- Support for time conditions

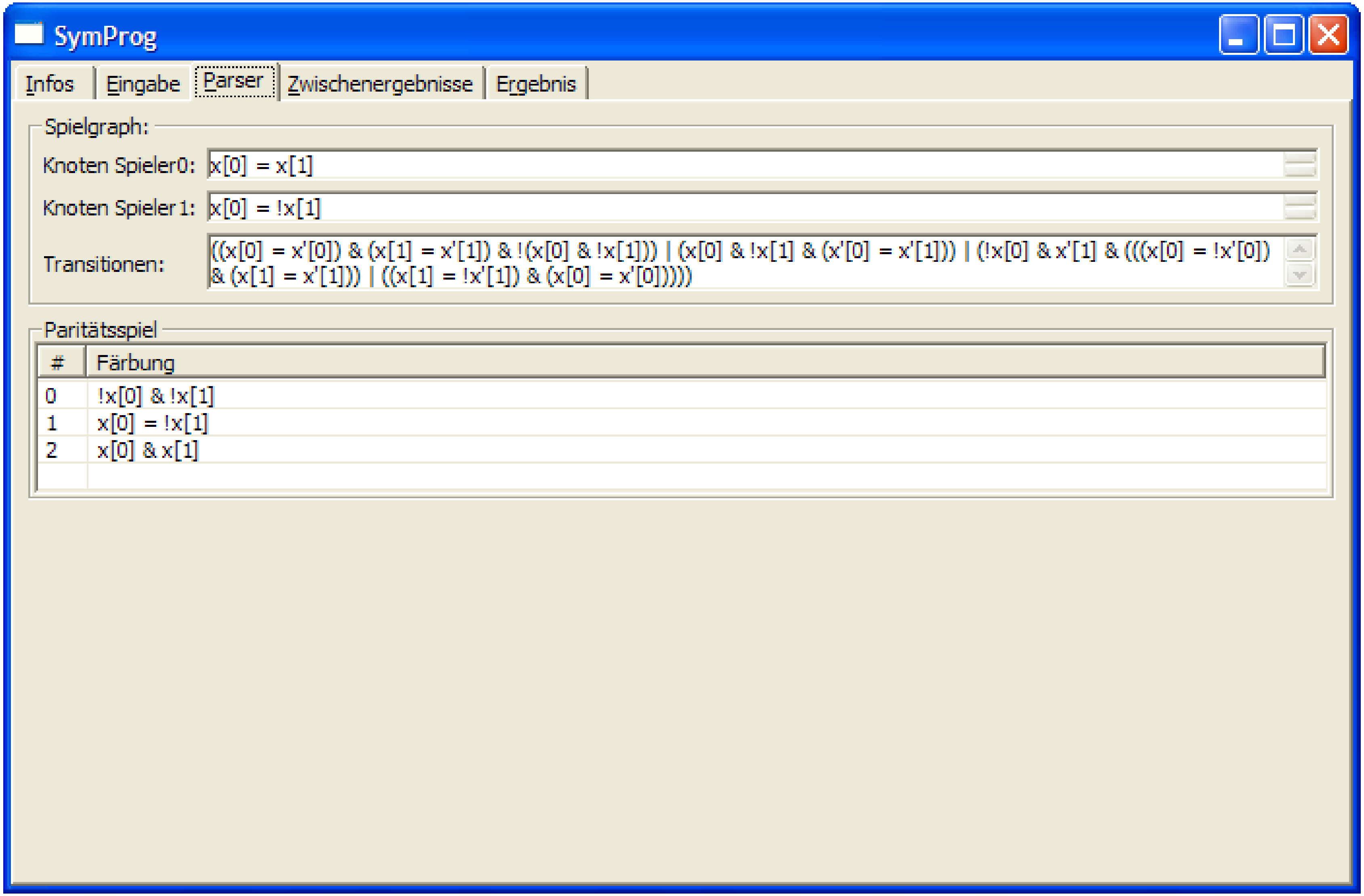
Screenshots



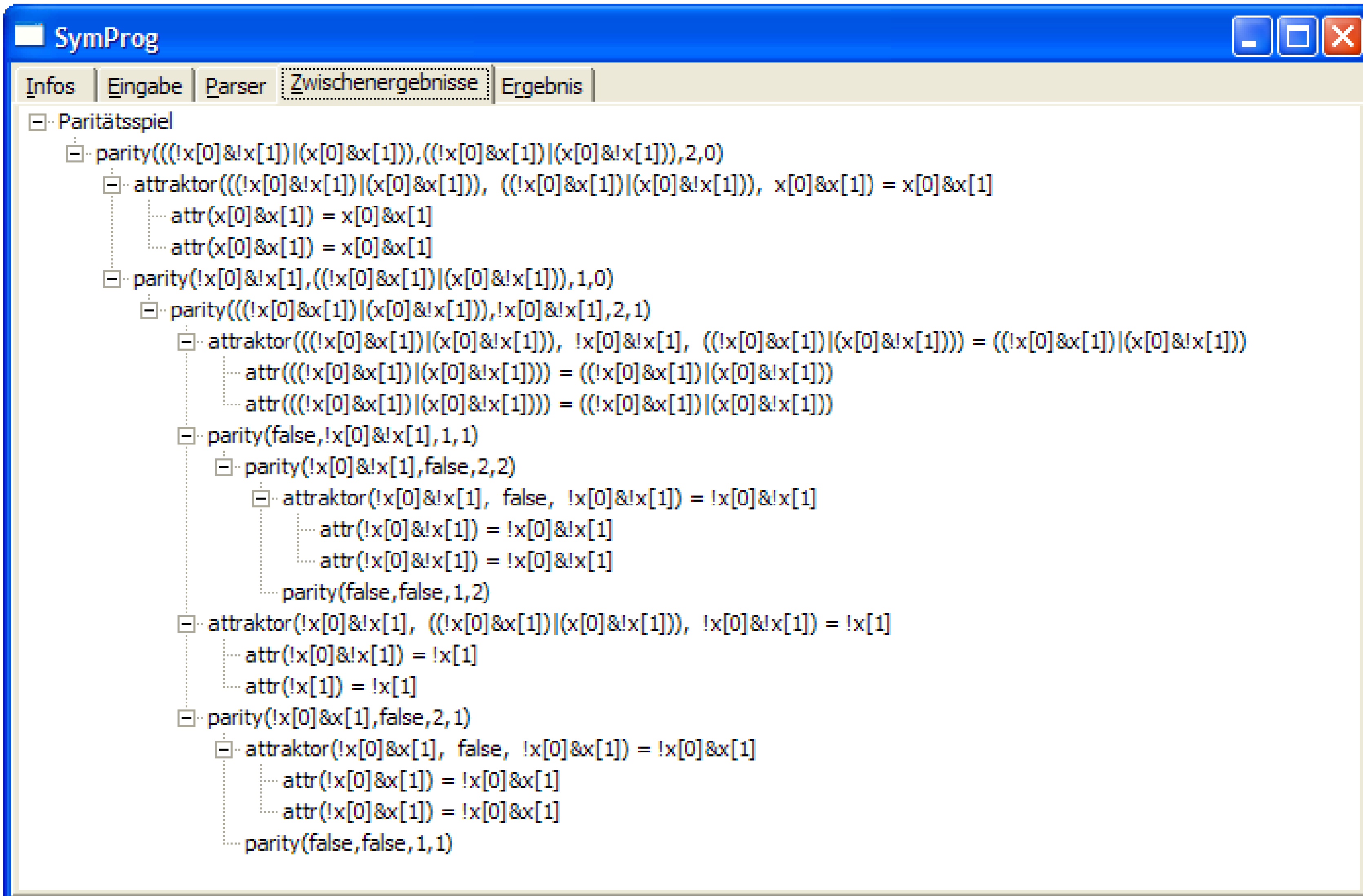
Screenshots - 2



Screenshots - 3



Screenshots - 4



Screenshots - 5