Optimizing Winning Strategies in Regular Infinite Games

SOFSEM 2008, January 2008

Wolfgang Thomas

RWTH AACHEN

A Quotation of 50 Years Ago

Alonzo Church

at the “Summer Institute of Symbolic Logic”

Cornell University, 1957:

“Given a requirement which a circuit is to satisfy, we may suppose the requirement expressed in some suitable logistic system which is an extension of restricted recursive arithmetic. The *synthesis problem* is then to find recursion equivalences representing a circuit that satisfies the given requirement (or alternatively, to determine that there is no such circuit).”

Alonzo Church (1903-1995)

APPLICATION OF RECURSIVE ARITHMETIC TO THE PROBLEM OF CIRCUIT SYNTHESIS

Alonzo Church

RESTRICTED RECURSIVE ARITHMETIC

Primitive symbols are individual (i.e., numerical) variables x, y, z, t,..., singulary functional constants $i_1, i_2, \ldots, i_\mu$, the individual constant 0, the accent ' as a notation for <u>successor</u> (of a number), the notation ( ) for application of a singulary function to its argument, connectives of the propositional calculus, and brackets [ ].

Axioms are all tautologous wffs. Rules are modus ponens; substitution for individual variables; mathematical induction,

$$\text{from } P \supset S^{a}_{a'}P| \quad \text{and} \quad S^{a}_{0}P| \quad \text{to infer } P;$$

and any one of several alternative recursion schemata or sets of recursion schemata.

$$\chi_1(x_1 + M + 1,\ 0, \cdots, 0,\ 0) \equiv \text{falsehood}$$

. . . .

. . . .

$$\chi_N(x_1 + M + 1,\ M, \cdots, M,\ g) \equiv \text{falsehood}$$

$$\chi_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, 0,\ 0) \equiv \text{falsehood}$$

. . . .

. . . .

$$\chi_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ 0) \equiv \text{falsehood}$$

$$\chi_2(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ 0) \equiv \text{falsehood}$$

. . . .

$$\chi_N(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ 0) \equiv \text{falsehood}$$

$$\chi_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ 1) \equiv \text{falsehood}$$

. . . .

. . . .

$$\chi_N(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ g) \equiv \text{falsehood}$$

$$\chi_1(0,\ 0, \cdots, 0,\ t + g + 1) \equiv \chi_1(0,\ 0, \cdots, 0,\ t + g) \vee Q_{100\cdots 0}[\chi_1(0,\ 0, \cdots, 0,\ t), \cdots, \chi_N(0,\ 0, \cdots, 0,\ t),\ \chi_1(0,\ 0, \cdots, 1,\ t), \cdots, \cdots, \chi_N(M+1,\ M+1, \cdots, M+1, t)$$

$$\chi_2(0,\ 0, \cdots, 0,\ t + g + 1) \equiv \chi_2(0,\ 0, \cdots, 0,\ t + g) \vee \overline{\chi}_1(0,\ 0, \cdots, 0,\ t + g) Q_{200\cdots 0}[\chi_1(0,\ 0, \cdots, 0,\ t), \cdots, \chi_N(0,\ 0, \cdots, 0,\ t),\ \chi_1(0,\ 0, \cdots, 1,\ t), \cdots, \cdots, \chi_N(M + 1,\ M + 1, \cdots, M + 1,\ t)]$$

. . . .

. . . .

$$\chi_N(M,\ M, \cdots, M,\ t + g + 1) \equiv \chi_N(M,\ M, \cdots, M,\ t + g) \vee \overline{\chi}_1(M,\ M, \cdots, M,\ t + g)\ \overline{\chi}_2(M,\ M, \cdots, M,\ t + g) \cdots \overline{\chi}_{N-1}(M,\ M, \cdots, M,\ t + g) Q_{NMM\cdots M}[\chi_1(0,\ 0, \cdots, 0, t), \cdots, \chi_N(0,\ 0, \cdots, 0,\ t),\ \chi_1(0,\ 0, \cdots, 1,\ t), \cdots, \cdots, \chi_N(2M + 1,\ 2M + 1, \cdots, 2M + 1, t)]$$

$$\chi_1(x_1 + M + 1,\ 0, \cdots, 0,\ t + g + 1) \equiv \chi_1(x_1 + M + 1,\ 0, \cdots, 0,\ t + g) \vee Q_{10\cdots 0}[\chi_1(x_1,\ 0, \cdots, 0,\ t), \cdots, \chi_N(x_1, 0, \cdots, 0,\ t),\ \chi_1(x_1,\ 0, \cdots, 1,\ t), \cdots, \cdots, \chi_N(x_1 + 2M + 2,\ M + 1, \cdots, M + 1,\ t)]$$

$$\chi_2(x_1 + M + 1,\ 0, \cdots, 0,\ t + g + 1) \equiv \chi_2(x_1 + M + 1,\ 0, \cdots, 0,\ t + g) \vee \overline{\chi}_1(x_1 + M + 1,\ 0, \cdots, 0,\ t + g) Q_{20\cdots 0}[\chi_1(x_1,\ 0, \cdots, 0,\ t), \cdots, \chi_N(x_1,\ 0, \cdots, 0,\ t),\ \chi_1(x_1,\ 0, \cdots, 1,\ t), \cdots, \cdots, \chi_N(x_1 + 2M + 2,\ M + 1, \cdots, M + 1,\ t)]$$

. . . .

. . . .

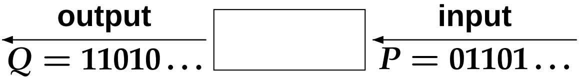
$$\chi_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g + 1) \equiv \chi_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g) \vee Q_1[\chi_1(x_1,\ x_2, \cdots, x_m,\ t), \cdots, \chi_N(x_1,\ x_2, \cdots, x_m,\ t),\ \chi_1(x_1,\ x_2, \cdots, x_m + 1,\ t), \cdots, \cdots, \chi_N(x_1 + 2M + 2,\ x_2 + 2M + 2, \cdots, x_m + 2M + 2,\ t)]$$

$$\chi_2(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g + 1) \equiv \chi_2(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g) \vee \overline{\chi}_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g) Q_2[\chi_1(x_1,\ x_2, \cdots, x_m,\ t), \cdots, \chi_N(x_1,\ x_2, \cdots, x_m,\ t),\ \chi_1(x_1,\ x_2, \cdots, x_m + 1,\ t), \cdots, \cdots, \chi_N(x_1 + 2M + 2,\ x_2 + 2M + 2, \cdots, x_m + 2M + 2,\ t)]$$

. . . .

$$\chi_N(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g + 1) \equiv \chi_N(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t + g) \vee \overline{\chi}_1(x_1 + M + 1,\ x_2 + M + 1, \cdots, x_m + M{+}1,\ t{+}g)\ \overline{\chi}_2(x_1{+}M{+}1,\ x_2 + M + 1, \cdots, x_m + M + 1,\ t{+}g) \cdots \overline{\chi}_{N-1}(x_1{+}M{+}1,\ x_2{+}M{+}1, \cdots, x_m + M + 1,\ t + g) Q_N[\chi_1(x_1,\ x_2, \cdots, x_m,\ t), \cdots, \chi_N(x_1,\ x_2, \cdots, x_m,\ t),\ \chi_1(x_1,\ x_2, \cdots, x_m + 1,\ t), \cdots,$$

Game-Theoretic View



For $t = 0, 1, 2, \ldots$: Input player (1) supplies bit $P(t)$, output player (2) responds by bit $Q(t)$

Bitstreams correspond to subsets of $\mathbb{N}$.

Use variables X, Y for subsets of $\mathbb{N}$.

Requirement $\varphi(X, Y)$ is considered as winning condition in an infinite two-person game.

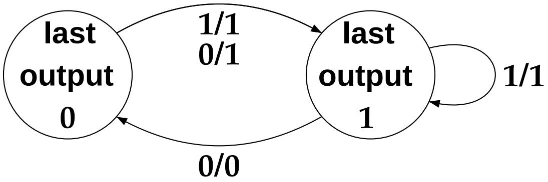
Play $\binom{P(0)}{Q(0)} \binom{P(1)}{Q(1)} \binom{P(2)}{Q(2)} \ldots$ is won by 2 if $(\mathbb{N}, \ldots) \models \varphi(P, Q)$

Example

$\varphi(X, Y)$:

- $\forall t\,(X(t) \rightarrow Y(t))$
- $\neg\exists t(\neg Y(t) \wedge \neg Y(t'))$
- $(\exists^\omega t\, \neg X(t) \rightarrow \exists^\omega t\, \neg Y(t))$

Solution:



This is a finite-state strategy (realized by a Mealy automaton).

Plan

1. The origin: Church’s Problem (done)
2. Muller games
3. Solving Muller games
4. Memory-optimal controllers
5. Optimal solutions for liveness requirements
6. Outlook

Muller Games

Approach for Solution of Church's Problem

1. Translation of formula φ into Muller automaton
2. Conversion of Muller automaton into a Muller game graph
3. Transformation of Muller game into parity game
4. Solution of parity game

Steps 1 and 2 go from logic to automata (and games).

Steps 3 and 4 show how to solve "regular infinite games".

Muller Automata

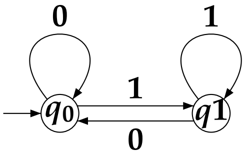
are finite automata $\mathcal{A} = (S, \Sigma, s_0, \delta, \mathcal{F})$ accepting ω-sequences.

Acceptance component: Family $\mathcal{F} = \{F_1, \ldots, F_k\}$ of state-sets.

$\mathcal{A}$ accepts $\alpha \Leftrightarrow$ the states occurring infinitely often in the run ρ of $\mathcal{A}$ on α form some set F_i

short: $\mathrm{Inf}(\rho) \in \mathcal{F}$

Example



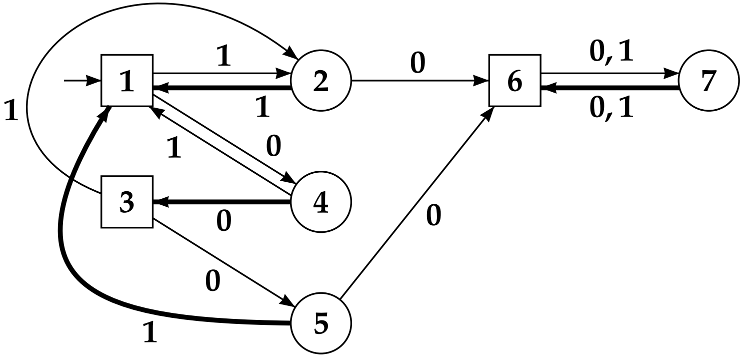
- with $\mathcal{F} = \{\{q_1\}\}$ accepts $(0+1)^*1^\omega$
- with $\mathcal{F} = \{\{q_1\}, \{q_0, q_1\}\}$ accepts $(0^*1)^\omega$

We dissolve a transition with $\binom{0}{1}$ into two transitions, marking that Player 1 picks 0 and Player 2 picks 1.

We obtain a "game graph".

Initial Example

$\varphi(X,Y)$: $\forall t\,(X(t) \longrightarrow Y(t)) \;\wedge\; \neg\exists t(\neg Y(t) \wedge \neg Y(t'))$
$\wedge\; (\exists^\omega t\, \neg X(t) \longrightarrow \exists^\omega t\, \neg Y(t))$



where $\mathcal{F} = \{\{1,2,3,4\},\{1,2,3,4,5\},\{1,3,4,5\}\}$

Game Graphs

A **game graph** **has the form** $G = (Q, Q_1, E)$ **where** $Q_1 \subseteq Q$ **and** $E \subseteq Q \times Q$ **is the transition relation satisfying**

$$\forall q \in Q : qE \neq \emptyset \quad \text{(i.e. } \forall q \exists q' : (q, q') \in E\text{)}$$

We set $Q_2 := Q \setminus Q_1$

A play is a sequence $\rho = r_0 r_1 r_2 \ldots$ **with** $(r_i, r_{i+1}) \in E$

Intuitively, a token is moved from vertex to vertex via edges, Player 1 / 2 deciding on the vertices of Q_1 **/** Q_2

Winning Conditions (Requirements)

in this talk:

- **Logical winning condition (e.g. written in MSO)**
- **Muller condition:**
  for play ρ: $\mathrm{Inf}(\rho) \in \mathcal{F}$
- **Weak Muller condition**
  for play ρ: $\mathrm{Occ}(\rho) \in \mathcal{F}$

Comparison with Church's Problem

1. Church's Problem uses a trivial graph (over $Q_1 = \{0, 1\}$ and $Q_2 = \{0', 1'\}$) and an MSO winning condition.
2. Model of reactive system: finite game graph and logical winning condition
3. Muller game: Finite game graph and Muller winning condition

Cases 1 and 2 reduce to case 3:

φ is equivalent to Muller automaton $\mathcal{A}_\varphi = (S, Q, s_0, \delta, \mathcal{F})$

Now take game graph over $Q \times S$ with Muller condition referring to second component.

Strategies

A strategy for player 2 from q is a function $f : Q^+ \to Q$, specifying for any play prefix $q_0 \ldots q_k$ with $q_0 = q$ and $q_k \in Q_2$ some vertex $r \in Q$ with $(q_k, r) \in E$

A strategy f for player 0 from q is called winning strategy for player 0 from q if any play from q which is played according to f is won by player 0 (according to the winning condition).

In the analogous way, one introduces strategies and winning strategies for player 1.

We say: Player 2 wins from q if s/he has a winning strategy from q

Winning Regions

For a game $\Gamma = (G, \varphi)$ with $G = (Q, Q_1, E)$, the **winning regions of players 1 and 2** are the sets

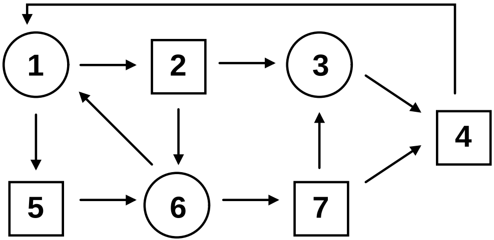
$$W_1 := \{q \in Q \mid \text{player 1 wins from } q\}$$

$$W_2 := \{q \in Q \mid \text{player 2 wins from } q\}$$

Remark: Each vertex q belongs at most to W_1 or W_2.

An Example

Example:



Winning condition for player 2: Vertex 3 should be reached.

Weak Muller game: Use $\mathcal{F} = \{F \mid 3 \in F\}$

$W_1 = \{1, 2, 4, 5, 6, 7\} \quad W_2 = \{3\}$

Determinacy

In general, the winning regions W_0, W_1 of players 1 and 2 satisfy $W_1 \cap W_2 = \emptyset$

A game is called determined if from each vertex either of the two players has a winning strategy.

Remark:

1. **There are (exotic) games which are not determined.**
2. **In descriptive set theory one investigates which abstract winning conditions define determined games.**
3. **All games in this talk determined.**
   (They are "Borel games".)

Church's Problem Reformulated

Given a game $\Gamma = (G, \varphi),\ G = (Q, Q_1, E)$

1. **Decide for each $q \in Q$ whether $q \in W_2$ (i.e. whether player 2 wins from q)**
2. **In this case:**
   Construct a suitable winning strategy from q (in the form of an automaton, or program)
3. **Optimize the construction of the winning strategy (e.g., time complexity) or optimize parameters of the winning strategy (e.g., size of memory).**

Solving a game means to provide algorithms for 1. and 2.

Special Strategies

If Q is finite, then a strategy is a word function $f : Q^+ \longrightarrow Q$

There are three basic types of strategies:

1. **computable (recursive),**
2. **finite-state (computable by a Mealy automaton)**
3. **positional (memoryless, value given by current vertex alone)**

Other types: pushdown strategy, counter strategy etc.

Büchi-Landweber Theorem

Finite Muller games are determined, one can compute the winning regions of the two players, and one can compute respective finite-state winning strategies.

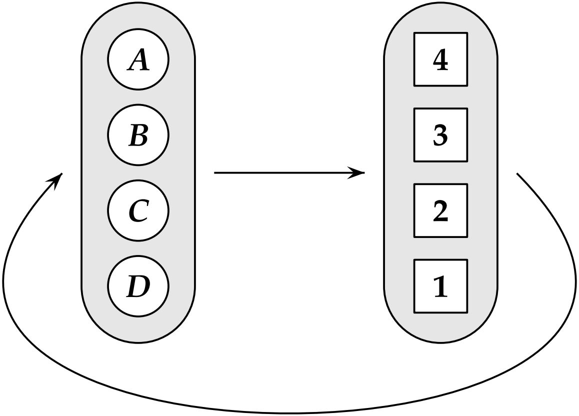
Construction of winning strategies is controller synthesis.

Finite-state controller synthesis is possible in automated manner for MSO- (or LTL-) specifications.

Solving Muller Games

An Interesting Muller Game (DJW-Game)

due to Dziembowski, Jurdziński, Walukiewicz (1997)



Number of letters chosen infinitely often should coincide with the highest number chosen infinitely often.

Latest Appearance Record

Visited letter	LAR
A	$ABCD$
C	$CA\underline{B}D$
C	$\underline{C}ABD$
D	$DCA\underline{B}$
B	$BDC\underline{A}$
D	$D\underline{B}CA$
C	$CD\underline{B}A$
D	$D\underline{C}BA$
D	$\underline{D}CBA$

Underlined position: “hit”

Example Scenario

Assume the states C and D are repeated infinitely often.

Then:

- **the states A and B eventually arrive at the last two positions and are not touched any more; so finally underlinings appear at most on positions 1 and 2**
- **position 2 is underlined again and again; if only position 1 is underlined from some point onwards, only the same letter would be chosen from there onwards (and not two states C and D as assumed)**

Solution of the DJW-Game

LAR-strategy for player 0:

During play, update and use the LAR as follows:

- **shift the current letter vertex to the front**
  underline the position from where the current letter was taken
- **move to the number vertex given by underlined position**

These are the two items performed by the strategy:

- **update of memory**
- **choice of next step ("output")**

Result: "Finite-state winning strategy" with $n! \cdot n$ states for a game graph with $2n$ vertices

Proof Strategy

Given a Muller game over G, the transition structure of the strategy automata can be constructed from $G = (Q, Q_1, E)$ alone:

- **Memory space: $\mathrm{LAR}(Q)$ (LAR's over Q)**
- **Memory-update during play $\rho \in Q^\omega$ according to LAR-update rule**
- **Missing item: Output function**

Core of Proof

For $\rho \in Q^\omega$ consider induced $\rho' \in \mathrm{LAR}(Q)$

$h :=$ maximal hit occurring infinitely often in ρ'

$R :=$ (eventually fixed) set up to this hit position h

Then: $\mathrm{Inf}(\rho) = R$

Reformulate winning condition using

$c : \mathrm{LAR}(Q) \rightarrow \{1, \ldots, 2 \cdot |Q|\}$

$c(\{q_{i_1}, \ldots, \underline{q_{i_h}}, \ldots, q_{i_n}) = 2h$ if $\{q_{i_1}, \ldots, q_{i_h}\} \in \mathcal{F}$, else $2h - 1$

Then: $\mathrm{Inf}(\rho) \in \mathcal{F}$ iff $\max(\mathrm{Inf}(c(\rho')))$ is even

This is the “parity condition”

On Parity Games

Emerson-Jutla and Mostowski (1991):

Parity games are determined (even over infinite game graphs), and on the winning region W_i Player i has a positional (!) winning strategy.

Proof by induction over the number of colors

Core of constrcution of winning strategy: Reachability analysis

Weak Muller Games

Winning condition: $\mathrm{Occ}(\rho) \in \mathcal{F}$

A strategy automaton needs only to remember which states have been visited.

Use "Appearance record" AR rather than LAR.

Introduce weak parity games, with winning condition

"the highest color of a visited vertex is even"

Memory states of strategy automata are sets of vertices rather than lists of vertices.

Looking Back

1. Translation of formula φ into Muller automaton
2. Conversion of Muller automaton into a Muller game graph
3. Transformation of Muller game into parity game
4. Solution of parity game

Current Developments

- **Generalizations of the game model:**
  Infinite-state, concurrent, stochastic, timed, weighted, distributed, multi-player games

- **Closer analysis (this talk)**
  1. **Memory-optimal controllers**
  2. **Optimal solutions for liveness requirements**

- **Other issues:**
  Definability of controllers
  Generalizing winning strategies

Memory-optimal Controllers

Memory Reduction

Fact: For a Muller game with n states
one can construct winning strategies with $n! * n$ states,
and $n!$ is also a lower bound.

But: There are two sources of memory:

- construction of Muller game arena
- construction of finite-state controller

Problem 1: How are these two steps related?

Problem 2: Understand the space of strategies

Three Approaches to Memory Reduction

- **Reduce memory for given strategy f**
  Use standard procedure as in DFA minimization

- **View the game graph as an automaton and reduce it first**
  (Holtmann, Löding (Aachen))

- **Search the space of all (winning) strategies to find one with minimal-memory implementation**
  (open problem, hint by Büchi-Landweber)

Holtmann-Löding Method

General plan:

- **Given a (weak) Muller game over Q,**
- **transform it into a (weak) parity game over $S \times Q$,**
- **Forgetting about the partition (Q_1, Q_2) we obtain an automaton with state-set S and input alphabet Q that accepts (with the (weak) parity condition) precisely the winning plays for Player 2.**
- **Main step: Mimimize / Reduce the size of this automaton in a way that a (weak) parity game over some $S_0 \times Q$ can be extracted.**
- **Use S_0 as memory space for winning strategy.**

Main Technical Points

- **Define $(s, q) \sim (s', q)$ iff from s with initial vertex q and from s' with initial vertex q the same plays are accepted.**
- **Define $s \equiv s'$ iff for all q we have $(s, q) \sim (s', q)$**
- **Then $\equiv$-classes can serve as new states.**
- **Use tests for $(s, q) \sim (s', q)$ (from ω-automata theory)**

Result: There are games with $c \cdot n$ vertices where the game graph reduction yields an exponential gain over the standard strategy minimization.

On the other hand, the approach misses some potentials of minimization and is not a complete method.

Optimal Solutions for Liveness Requirements

Optimality in Request-Response Games

Game arena $G = (V, V_0, E)$

Subsets $Rqu_1, \ldots, Rqu_k \subseteq V$**: "Requests"**

Subsets $Rsp_1, \ldots, Rsp_k \subseteq V$**: "Responses"**

RR-condition:

$$\bigwedge_{i=1}^{k} \forall s (Rqu_i(s) \longrightarrow \exists t \; (s < t \wedge Rsp_i(t)))$$

LTL:

$$\bigwedge_{i=1}^{k} \mathbf{G}(Rqu_i \longrightarrow \mathbf{XF}\; Rsp_i)$$

Standard Solution of RR-Games

- **It suffices to keep a memory for the set of "open requests"**
  Memory size: 2^k for k conditions
- **Reduction to Büchi games**
- **Result: Winning strategy which ensures bounded waiting time between request and response**
  (Bound $B := k \cdot |V|$).

Problem: Use finer measure than maximum of waiting times

Measuring Quality of Solution

Penalty function associates to i-th moment of waiting a penalty

- **Linear Penalty model:**
  For each moment of waiting (for each RR-condition) pay 1 unit
- **Quadratic Penalty model:**
  For the i-th moment of waiting pay i units
- **More general, use strictly growing unbounded penalty function**

Activation of i-th condition in a play ϱ is a visit to Rqu_i such that all previous visits to Rqu_i are already matched by an Rsp_i-visit.

Values of Plays and Strategies

For a given penalty function define:

- $w_\varrho(n) =$ **sum of penalties in** $\varrho(0)\ldots\varrho(n)$ **divided by number of activations**

  "average penalty sum per activation"

- $w(\varrho) = \limsup_{n\to\infty} w_\varrho(n)$

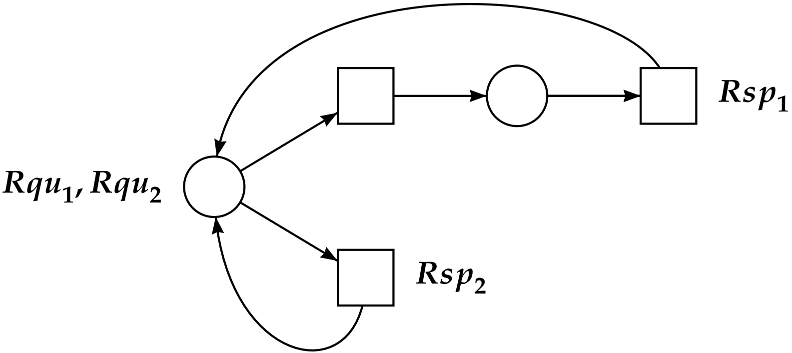
Given a strategy σ **for controller and a strategy** τ **for adversary**

- $\varrho(\sigma,\tau) :=$ **the play induced by** σ **and** τ
- $w(\sigma) := sup_\tau\, w(\varrho(\sigma,\tau))$

Call σ **optimal if there is no other strategy with smaller value.**

On the Linear Penalty

For the linear penalty model, a finite-state optimal strategy does not exist in general:



Theorem

(with F. Horn and N. Wallmeier)

For any strictly increasing unbounded penalty function one can decide whether a RR-game is won by controller and in this case one can compute a finite-state optimal winning strategy.

Proof ingredients:

- **It suffices to consider strategies with value $\leq M$ (induced by bounded waiting time of standard solution).**
- **Conversely: For strategies with value $\leq M$ one can assume bounded waiting time.**
- **Reduction to mean-payoff games (Zwick-Paterson)**

Building a Mean-Payoff Game

From a game graph $G = (V, E)$ with k conditions

proceed to a game graph over $V \times \mathbb{N}^k$

State format: $(v, n_1, \ldots, n_k)$

$n_i =$
current waiting time for i-th condition since last activation

Derived mean-payoff game:

For each edge $e \; = \; (u, \overline{m}) \longrightarrow (v, \overline{n})$

introduce edge weight

$w(e) = n_1 + \ldots + n_k$ (sum of current penalties)

Boundedness Lemma

Let σ be a winning strategy of value $\leq M$

Then one can construct a winning strategy σ' with bounded waiting times such that $w(\sigma') \leq w(\sigma)$.

Consequence:

In the mean-payoff game, it suffices to consider waiting time vectors in a domain $[0, B]^k$ rather than $\mathbb{N}^k$.

So we obtain a finite MPG which can be solved.

Intuition for Boundedness Lemma

Example scenarium:

Consider a winning strategy σ of value $\leq M$ which allows unbounded waiting times just for the last RR-condition.

States: $(v, \overline{m}, m)$ with $v \in V, \overline{m} \in [0, B]^{k-1}, m \geq 0$

In a play with unbounded waiting times for the last condition, pick a "critical segment" $(v, \overline{m}, m), \ldots, (v, \overline{m}, m')$ where each position has a penalty $\geq M$.

In σ', this play segment is skipped. This decreases

- **the waiting time for the last component**
- **the value of the strategy (each deleted step has $\geq$ average penalty)**

Outlook

Broader View on Transformations

A strategy defines an operator $T : \{0,1\}^\omega \longrightarrow \{0,1\}^\omega$

T is continuous if $T(P)(t)$ depends only on a finite segment of P.

The MSO-condition $\varphi(X, Y)$:

$$(\exists t X(t) \leftrightarrow \forall t Y(t)) \quad \wedge \quad (\forall t Y(t) \vee \forall t \neg Y(t))$$

is solvable only by the non-continuous operator T_0 with

$T_0(\emptyset) = 0^\omega$ and $T_0(P) = 1^\omega$ for $P \neq \emptyset$

On Continuity

Landweber, Hosch (1971): It is decidable whether a MSO winning condition can be solved by a finite-state strategy with bounded delay.

Example 1: Division of a sequence by two

T^{-}: $P(0)P(1)\ldots \mapsto P(0)P(2)P(4)\ldots$

T^{-} **is continuous, with linearly increasing delay.**

Example 2: Doubling a sequence

T^{+}: $P(0)P(1)\ldots \mapsto P(0)P(0)P(1)P(1)\ldots$

T^{+} **is bit-by-bit-computable with unbounded memory, or with a sequential machine.**

Conclusion

- Church's Problem is far from closed.
- A current challenge is to shift the investigation from decision problems to various optimization problems.
- Another challenge is to investigate the synthesis of generalized automata (e.g., sequential machines)

General Sources

- D. Perrin, J.E. Pin, Infinite Words, Elsevier 2004
- W. Th., Languages, Automata and Logic, in Handbook of Formal Languages, Vol. 3, Springer 1997
- W. Th., Church's Problem and a Tour Through Automata Theory, in: Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday (A. Avron, N. Dershowitz, A. Rabinovich, eds.), LNCS, Springer 2008.