What are “Good” Strategies in Infinite Games?

Wolfgang Thomas

RWTH AACHEN

Workshop on Strategic Reasoning, Grenoble, April 2014

Infinite Games: Three Stages

- **Descriptive set theory (1920's ff.): Existence of strategies**
- **Automata theory (1960's ff.): Construction of strategies executable by automata**
- **Current research: Efficient construction of "good" or even "optimal" strategies**

A strategy may be called “good” if . . .

- it is sparse in memory,
  i.e., implementable by an automaton with few states
- it is responsive
  i.e., serves requests quickly,
- it is permissive
  i.e., allows several choices (nondeterministically),
- it is implementable by a small program,
- it is definable with weak logical means.

Plan

1. **Background: Regular infinite games and their solution**

2. **Optimizing strategies**
   - **Memory size**
   - **Mean-payoff games and optimizing parameters of behaviour**

3. **Giving strategies a good format**
   - **Strategy machines**
   - **Boolean programs**
   - **Logical descriptions**

Background: Regular Infinite Games

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,···,i_μ, 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,

from $P \supset S^a_{a'}P|$ and $S^a_0P|$ to infer P;

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

Alonzo Church

A Citation

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).”

(By “circuits”, Church means finite automata with output.)

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

. . . . . .

. . . . . .

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

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

. . . . . .

. . . . . .

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

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

. . . . . .

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

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

. . . . . .

. . . . . .

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

$\chi_1(0,\ 0,\cdots,0,\ t + g + 1) \equiv \chi_1(0,\ 0,\cdots,0,\ t + g)\ \vee$
$Q_{100\cdots0}[\chi_1(0,\ 0,\cdots,0,\ t),\cdots,\chi_N(0,\ 0,\cdots,$
$0,\ t),\ \chi_1(0,\ 0,\cdots,1,\ t),\cdots,\cdots,\chi_N(M+1,\ M+1,\cdots,M+1,t)$

$\chi_2(0,\ 0,\cdots,0,\ t + g + 1) \equiv \chi_2(0,\ 0,\cdots,0,\ t + g)\ \vee$
$\overline{\chi}_1(0,\ 0,\cdots,0,\ t + g)Q_{200\cdots0}[\chi_1(0,\ 0,\cdots,0,\ t),\cdots,$
$\chi_N(0,\ 0,\cdots,0,\ t),\ \chi_1(0,\ 0,\cdots,1,\ t),\cdots,\cdots,$
$\chi_N(M + 1,\ M + 1,\cdots,M + 1,\ t)]$

. . . . . .

. . . . . .

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

$\chi_1(x_1 + M + 1,\ 0,\cdots,0,\ t + g + 1) \equiv \chi_1(x_1 + M + 1,\ 0,$
$\cdots,0,\ t + g)\ \vee\ Q_{10\cdots0}[\chi_1(x_1,\ 0,\cdots,0,\ t),$
$\cdots,\chi_N(x_1,0,\cdots,0,\ t),\ \chi_1(x_1,\ 0,\cdots,1,\ t),\cdots,\cdots,$
$\chi_N(x_1 + 2M + 2,\ M + 1,\cdots,M + 1,\ t)]$

$\chi_2(x_1 + M + 1,\ 0,\cdots,0,\ t + g + 1) \equiv \chi_2(x_1 + M + 1,\ 0,\cdots,0,$
$t + g)\ \vee\ \overline{\chi}_1(x_1 + M + 1,\ 0,\cdots,0,\ t + g)Q_{20\cdots0}[\chi_1(x_1,$
$0,\cdots,0,\ t),\cdots,\chi_N(x_1,\ 0,\cdots,0,\ t),\ \chi_1(x_1,\ 0,\cdots,$
$1,\ t),\cdots,\cdots,\chi_N(x_1 + 2M + 2,\ M + 1,\cdots,M + 1,\ t)]$

. . . . . .

. . . . . .

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

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

. . . . . .

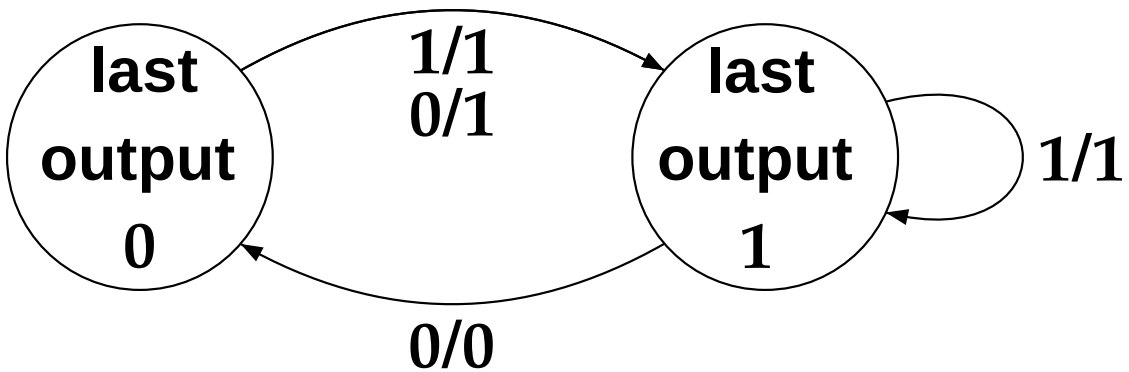
$\chi_N(x_1 + M + 1,\ x_2 + M + 1,\cdots,x_m + M + 1,\ t + g + 1) \equiv$
$\chi_N(x_1 + M + 1,\ x_2 + M + 1,\cdots,x_m + M + 1,\ t + g)\ \vee$
$\overline{\chi}_1(x_1 + M + 1,\ x_2 + M + 1,\cdots,x_m + M+1,\ t+g)\ \overline{\chi}_2(x_1+M+1,$
$x_2 + M + 1,\cdots,x_m + M + 1,\ t+g)\cdots\overline{\chi}_{N-1}(x_1+M+1,\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

Example

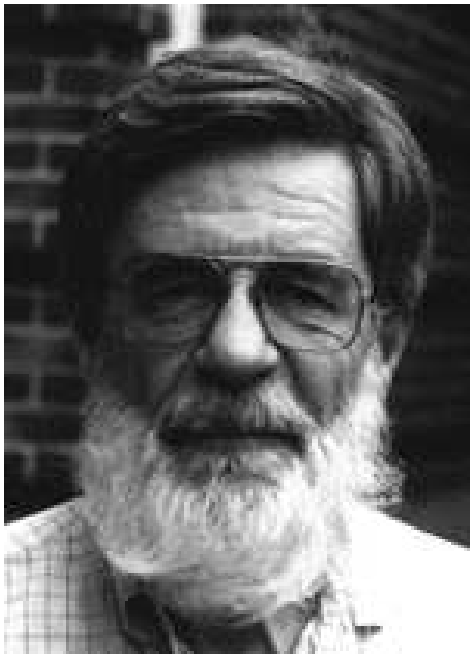
Requirement:

1. $\forall t\ X(t) = 1 \ \rightarrow \ Y(t) = 1$
2. $\neg \exists t\ Y(t) = Y(t+1) = 0$
3. $\exists^{\omega} t\ X(t) = 0 \ \rightarrow \ \exists^{\omega}\ t\ Y(t) = 0$

A solution:



Classical Methodology for Synthesis

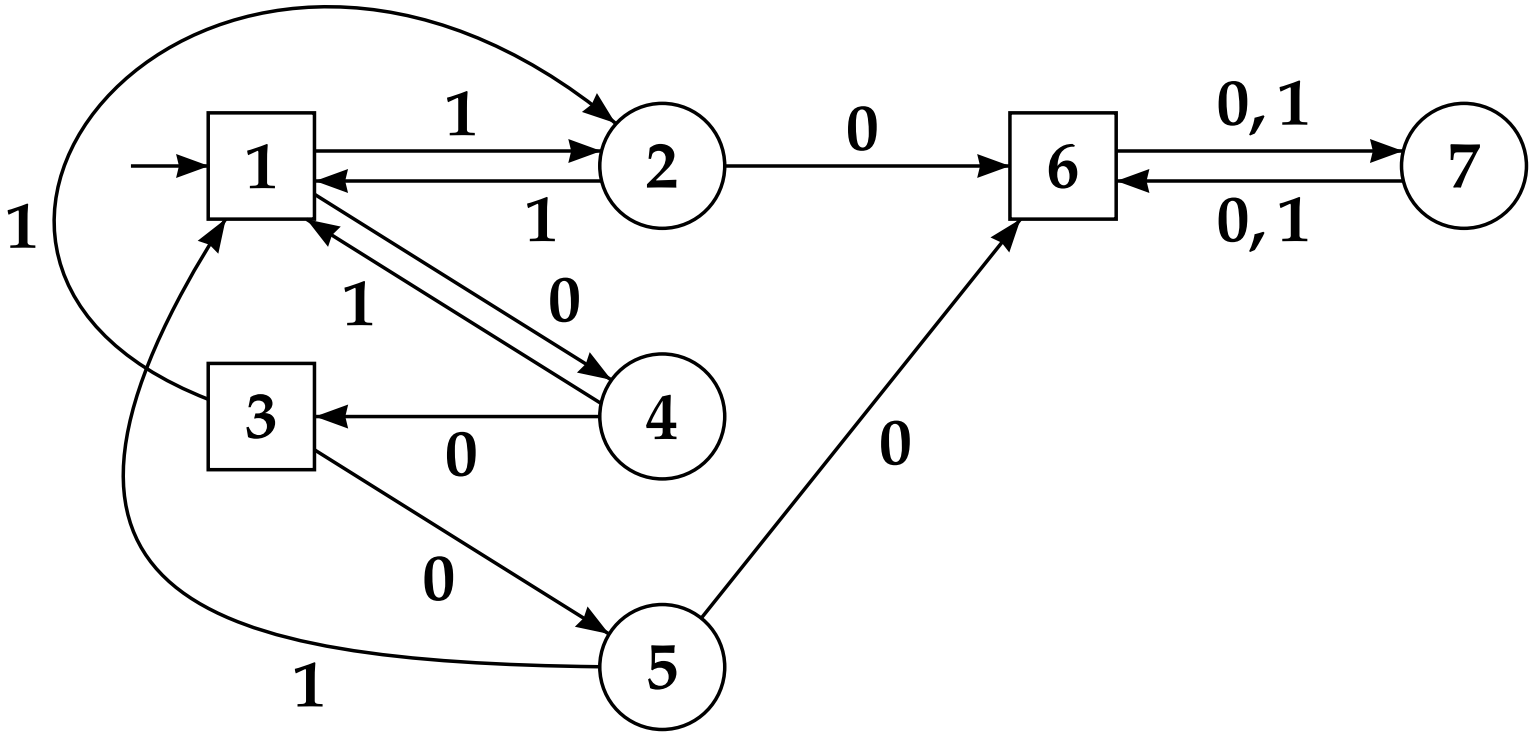


R. McNaughton

Given a requirement φ (in MSO-logic), understand it as a definition of a two-person game between Players 1 and 2:

- **Transform φ into a finite game arena with Muller winning condition, and solve this "Muller game".**

Example of a Muller Game



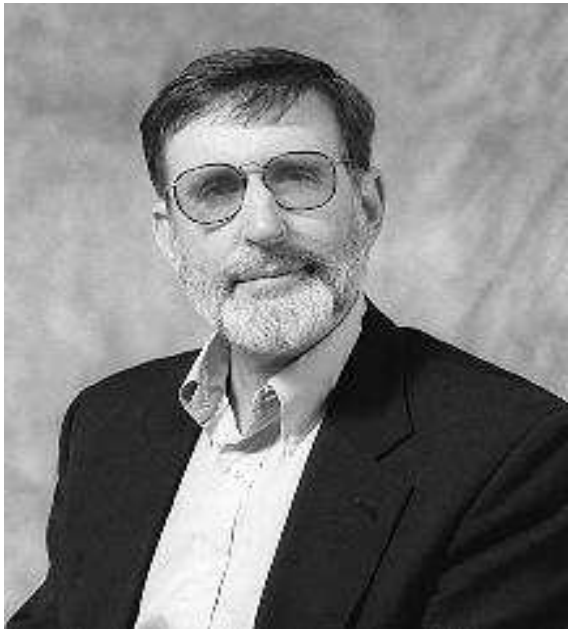
Player 2 wins if the infinitely often visited states are:

$\{1,2,3,4\}$ **or** $\{1,2,3,4,5\}$ **or** $\{1,3,4,5\}$ **or** $\{1,2\}$ **or** $\{1,4\}$

Format of Muller game: $G = (Q, Q_1, Q_2, E)$ **with** $\mathcal{F} \subseteq 2^Q$

Büchi-Landweber Theorem

J.R. Büchi



L.H. Landweber

For a Muller game over a finite arena with designated start vertex

- **one of the two players has a winning strategy,**
- **one can decide who wins,**
- **one can construct a finite-state strategy for the winner.**

Optimizing Memory Size

A Guiding Example: The DJW Game

due to Dziembowski, Jurdzinski, Walukiewicz (LICS 1997)

Player 1 picks letters $\{A, B, C, D\}$

Player 2 picks numbers from $\{1, 2, 3, 4\}$

Winning condition for Player 2:

- **Number of infinitely often chosen letters = Highest number infinitely often chosen**

LAR: Latest Appearance Record

D	B	B	D	C	B	A	B	A	C	B	A	B	A	C
A	D	B	$\underline{B}$	D	C	B	A	B	A	C	B	A	B	A	C
B	A	D	D	$\underline{B}$	D	C	B	$\underline{A}$	$\underline{B}$	A	C	B	$\underline{A}$	$\underline{B}$	A
C	B	$\underline{A}$	A	A	B	$\underline{D}$	C	C	C	$\underline{B}$	$\underline{A}$	$\underline{C}$	C	C	$\underline{B}$
$\underline{D}$	$\underline{C}$	C	C	C	$\underline{A}$	A	$\underline{D}$	D	D	D	D	D	D	D	D

Discussion

- Player 2 (choosing numbers) wins the DJW game: Keep the LAR during the play and always go to the number indicated by the underlining
- Over an arena with $O(n)$ states we need a memory of factorial size
- DJW: This game cannot be solved with less than $n!$ states.
- The strategy automaton realizes a huge case distinction - is there a representation which exploits the simple way LAR's are updated?
- Minimization of a given finite state strategy $\mathcal{S}$ is easy (classical DFA mimimization).
- But there may be other strategies with less states, not obtainable from $\mathcal{S}$.

A Challenging Problem of 1969

$Zt' = H[Xt, Yt, Zt]$.

(α) If $Zt' \in \{s_1, \ldots, s_n\}$, let i be the first such that $Zt' = s_i$. Then,
$Vt' = [A_1, s_1, h_1, \ldots, A_i, A_i(s_i), h_i]$
$kt' = h_1$.

(13) (β) If $Zt' \in P_{kt-1}[A_1, s_1, \ldots, A_n, s_n]$ but not (α), let j be the first such that $Zt' \in A_j \cap R_{kt-1}[A_1, s_1, \ldots, A_j, s_j]$ ($Zt' \in R_{kt-1}[\ \ ]$ if $j = 0$). Then,
$Vt' = [A_1, s_1, h_1, \ldots, A_j, s_j, h_j]$
$kt' = kt-1$.

(γ) If $Zt' \in Q_{kt-1}[A_1, s_1, \ldots, A_n, s_n]$ but neither (α) nor (β), let B be the first in the chosen order of U such that, $Zt' \in B \subset A_n$ and $\bigwedge_{u \in B} u \in R_{kt-1}[A_1, s_1, \ldots, A_n, s_n, B, B(u)]$. Then,
$Vt' = [A_1, s_1, h_1, \ldots, A_n, s_n, h_n, B, B(Zt'), kt-1]$
$kt' = kt-1$.

PROBLEM. Modify the recursions (13) to a schema with parameters which, by proper additional specifications for the parameters, will yield any given deterministic operator which solves $\mathfrak{C}(X, Y)$ for Y. Do the same for (16) and solutions of $\mathfrak{C}(X, Y)$ for X.

Optimizing Behaviour of Strategies

Example: Request-Response Games

Game arena $G = (Q, Q_1, Q_2, E)$

Subsets $Rqu_1, \ldots, Rqu_k \subseteq Q$: **“Requests”**

Subsets $Rsp_1, \ldots, Rsp_k \subseteq Q$: **“Responses”**

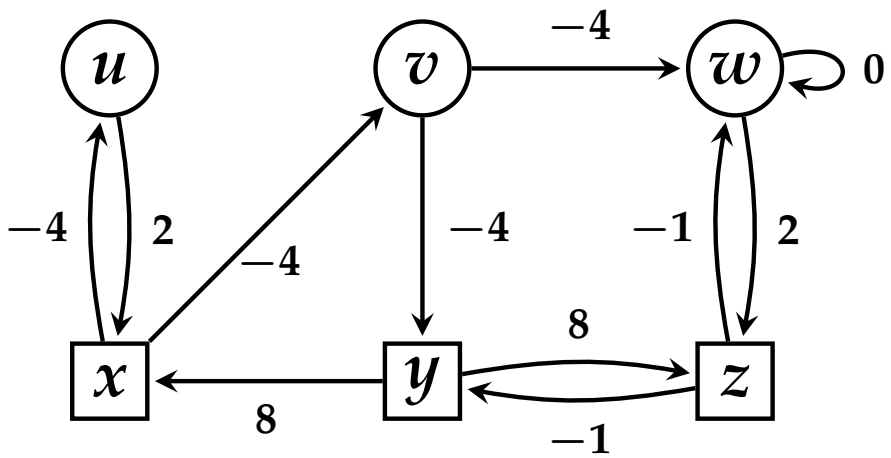
Format of winning condition:

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

Measure the quality of a winning strategy in terms of the waiting times it induces.

Penalty model: For the i-th moment of waiting pay i units

Mean Payoff Games



Value of a finite play $q_0 \cdots q_n$:

$$\frac{1}{n} \cdot \sum_{i=0}^{n-1} r(q_i, q_{i+1})$$

In the limit, Player 1 tries to minimize and Player 2 tries to maximize this value.

More Formally

A **mean payoff game** is of the form $\mathcal{G} = (Q, Q_1, Q_2, E, r)$ where (Q, Q_1, Q_2, E) is a finite game arena and

$$r : E \to \mathbb{Z}$$

is a function assigning a reward to each edge.

The players build up a play $\varrho = q_0 q_1 q_2 \cdots$ where

- Player 1 tries to minimize
$$r_1(\varrho) := \limsup_{n \to \infty} \left(\frac{1}{n} \cdot \sum_{i=0}^{n-1} r(q_i, q_{i+1}) \right)$$
- Player 2 tries to maximize
$$r_2(\varrho) := \liminf_{n \to \infty} \left(\frac{1}{n} \cdot \sum_{i=0}^{n-1} r(q_i, q_{i+1}) \right)$$

Positional Determinacy of Mean Payoff Games

Ehrenfeucht, Mycielski (1979) and Zwick, Paterson (1996):

For each finite mean payoff game there are positional strategies τ^* of Player 1 and σ^* of Player 2 such that for each vertex q

$$\inf_\tau \sup_\sigma r_1(\varrho_{\tau,\sigma,q}) = \sup_\sigma r_1(\varrho_{\tau^*,\sigma,q})$$
$$= \inf_\tau r_2(\varrho_{\tau,\sigma^*,q}) = \sup_\sigma \inf_\tau r_2(\varrho_{\tau,\sigma,q})$$

The problem of determining whether for given G and q the value of q is > 0 is in NP $\cap$ co-NP.

Back to Request-Response Games

For each RR-pair give penalty i for the i-th moment of waiting

- **Penalty of play prefix $\varrho(0) \ldots \varrho(n)$ is the sum of penalties so far, divided by number of "activations".**
- **Penalty of play ϱ is the limes superior of the play prefix penalties.**
- **Penalty value of strategy σ is the supremum over the penalties of all plays compatible with σ.**
- **Call σ optimal if there is no other strategy with smaller penalty value.**

Optimal Solution of RR-Games

One can decide whether a RR-game is won by controller (Player 2) and in this case one can compute a finite-state optimal winning strategy.

(Horn, Ths., Wallmeier, Zimmermann 2014)

Proof ingredients:

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

Boundedness Lemma

Let σ be a winning strategy in an RR-game with bounded waiting times.

Then there is a bound B for waiting times such that any τ with value(τ) $\leq$ value(σ) respects B.

Consequence:

We can build a finite-state mean payoff game to compute an optimal strategy.

Drawback: B is doubly exponential in the product of arena size n and number k of RR-conditions.

Building a Mean Payoff Game

For an arena $G = (Q, Q_1, Q_2, E)$ with k conditions

construct a new arena over $Q \times [0, \ldots, B]^k$

with states $(q, n_1, \ldots, n_k)$ where

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

Derived mean-payoff game:

For each edge $e = (p, \overline{m}) \to (q, \overline{n})$ introduce weight

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

Permissive Strategies

Model: Game on finite graph with the parity winning condition.

After a finite play prefix, a strategy just blocks certain edges.

Idea:

- **A strategy should narrow the system's behaviour as little as possible.**
- **This supports modular constructions: Adding requirements leads to a refinement of strategies.**

New Penalties

For strategy σ (of Player 2):

- **After a play prefix $\varrho(0)\ldots\varrho(n)$ it is the total number of edges blocked by σ up to time n.**
  Call this $\pi_\sigma(\varrho(0)\ldots\varrho(n))$.
- **Penalty of complete play ϱ is the lim sup over the average values $\frac{1}{n}\cdot\pi_\sigma(\varrho(0)\ldots\varrho(n))$**
- **Penalty associated with strategy σ:**
  supremum of penalties of plays consistent with σ.
- **Permissiveness value of game = infimum of this value over all winning strategies of Player 2.**

Result

- For parity games on finite graphs, the permissiveness value is computable.
- There are games where this value is not realizable by a finite-state strategy.

(Bouyer, Markey, Olschewski, Ummels 2010)

Open: Finite presentation of more general strategies such that computability of most permissive strategies is possible.

Giving Strategies a Good Format

A Selection

- Transition systems
- Circuits
- Symbolic presentations (Bloem at al., Ehlers)
- Boolean programs (Madhusudan, Brütsch)
- Strategy machines (Gelderie)
- Hierarchical transition systems (Aminof, Mogavero, Murano)
- Strategies composed from components of a “component library” (Lustig, Vardi)
- Logical formulas (Rabinovich, Th., Selivanov, . . .)

Another Quote from 1969

We have not seriously investigated whether the algorithms of Theorems 3 and 6 can be improved to a point of usefulness in the deisgn of sequential circuits. As they include conversion of propositional formulas into normal form, it seems that presently available computing equipment could not carry a significant part of our algorithms. Nevertheless, our solution automata of §5, like the construction of [14], provide examples of strictly finite devices which accomplish surprisingly intricate tasks.

Strategy Machines

Strategy Machines

are Turing machines with designated input and output states q_I and q_O and three kinds of tapes:

- **IO-tape**
- **memory tape**
- **k computation tapes**

An iteration:

(1) Receive input onto IO-tape (overwriting previous content) in state q_I, with empty computation tapes

(2) compute an output using the computation tapes and the memory tape

(3) write output on IO-tape and move to state q_O

Refined Complexity Measures

- **Latency**: Time of an iteration (between leaving q_I and entering q_O)
- **Space consumption**: # of memory-tape-cells visited during any previous iteration
- **Size**: Number of control states

A strategy machine can restrict to necessary information; it adapts to the play and converges towards a winning strategy.

This leads to small strategy machines for Muller and Streett games.

Muller and Streett Games

For a Muller game over G with family $\mathcal{F}$ of winning loops of Player 2, a strategy machine can be built whose size, space consumption, and latency are all polynomial in $|G| + |\mathcal{F}|$ if Player 2 wins this game.

An analogous result holds for Streett games.

(Gelderie, MFCS 2012)

Boolean Programs

Definion by Example

Boolean variables $B = \{b_1, b_2, b_3\}$

input b_1;
input b_2;
if b_1 then output b_2 else $b_1 := b_2$;
while true do
    input b_2;
    $b_3 := b_1 \wedge b_2$;
    output b_3

Synthesis (Madhusudan)

- Given regular specification $R \subseteq (\{0,1\} \times \{0,1\})^\omega$,
- consider a Büchi automaton $\mathcal{A}$ for the complement of R
- introduce a finite set of Boolean variables B.
- construct a tree automaton $\mathcal{B}$ that given a program parse tree t does the following, walking down and up the tree:
- It guesses an input bit sequence, determines step by step the output bit sequence using t, and simultaneously simulates $\mathcal{A}$

$\mathcal{B}$ recognizes a regular set of trees: those programs that violate the specification.

The complement tree language $\overline{T(\mathcal{B})}$ contains all correct programs.

Discussion

Pros

- We get all correct programs and thus also “minimal” ones (e.g. where the parse tree has minimal height),
- procedure is independent of the syntax of the specification,
- the complexity is exponential in the size of $\mathcal{A}$,
- the procedure can be adapted to handle delays in the input-output sequence (Brütsch 2013).

Cons

- The procedure works for fixed B; it needs a guess how many Boolean variables suffice.
- This number may be large: For LTL specifications of length n it may be $2^{\sqrt{n}}$ (Brütsch I& C to appear).

Logical Description of Strategies

Another Citation of 1969

More generally, the following type of game problem is naturally suggested by automata theory. Given a class of games G: (1) can one effectively decide, for any $\mathfrak{G} \in G$, which player has a winning strategy? (2) Just how simple winning strategies do exist for games in G? For example, is there a recursive or even a finite automata winning strategy for $\mathfrak{G} \in G$?

Strategies and Definability

A strategy for Player 2 is a map

$$\binom{\alpha(0)}{\beta(0)}\ \binom{\alpha(1)}{\beta(1)}\ \ldots\ \binom{\alpha(k)}{*} \quad \mapsto \quad 0/1$$

Strategies can be defined by sentences interpreted over such finite play prefixes,

in the sense that the truth value is the bit to be chosen.

General Problem: Relate the logic for describing winning strategies to the logic used to define the game.

More Precisely

A strategy $f : \binom{P(0)}{Q(0)} \binom{P(1)}{Q(1)} \dots \binom{P(k-1)}{Q(k-1)} \binom{P(k)}{*} \mapsto$ **0/1**

is MSO-definable iff there is an MSO-formula $\psi(X, Y, z)$ **such that**

$$([0,k], <) \models \psi((P \cap [0,k]),\ (Q \cap [0,k-1]),\ k)$$

iff

$$f(\binom{P(0)}{Q(0)} \dots \binom{P(k-1)}{Q(k-1)} \binom{P(k)}{*}) = 1$$

$\mathcal{L}$-Definable Games and Strategies

An $\mathcal{L}$-defined game is determined with $\mathcal{L}'$-definable strategies if

for each $\mathcal{L}$-formula $\varphi(X, Y)$, there is either an $\mathcal{L}'$-definable winning strategy of Player 1 or an $\mathcal{L}'$-definable winning strategy for Player 2.

Büchi-Landweber:

MSO-defined games are determined with MSO-definable strategies.

Some Results

- Games definable in FO-logic (over $(\mathbb{N}, <)$) are determined with strategies definable in the same logic.
  (Selivanov 2006, Rabinovich, Th. 2007)
- This fails for the levels $B(\Sigma_n)$ of the quantifier alternation hierarchy within FO-logic over $(\mathbb{N}, <)$:
- Winning strategies need an increase of quantifier alternation rank by at least 1, and an increase by 2 suffices. (Chaturvedi, Olschewski, Th. 2011)
- For games defined in Presburger arithmetic, even full first-order arithmetic is insufficient for defining winning strategies.

Presburger Arithmetic

A winning condition $\varphi(X,(Y,Z))$ can fix that $Z =$ Squares:

$0 \in Z \wedge \forall x_1, x_2, x_3 (x_1 < x_2 < x_3$ **successive in** Z
$\rightarrow\ x_3 - x_2 = (x_2 - x_1) + 2)$

Putnam 1957: In FO$(+, \mathrm{Squ})$ multiplication is definable.

Proof: $2xy = (x+y)^2 - x^2 - y^2$

$x^2 = y \Leftrightarrow$
$y \in \mathrm{Squ}\ \wedge\ y - (2x-1)$ **is the greatest square** $< y$

Consequence: Each winning condition $\exists^\omega x R(X,(Y,\mathrm{Squ}),x)$ with recursive R can be expressed.

Even hyperarithmetical winning strategies do not suffice (but the winning conditions are arithmetical).

Conclusion

Three Problems

- Find ways to systematically explore the space of winning strategies
- Make strategy optimization practical and understand how to handle (the trade-offs between) multiple optimization criteria.
- Develop a compositional framework of strategy construction which reflects the structure of the (logical) specifications.