

Church's Problem after 50 Years

$L_S, V \models 10Y$, Cachan, November 2007

Wolfgang Thomas

RWTHAACHEN



Alonzo Church (1903-1995)

Church's Problem

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).”

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, \dots , singular functional constants i_1, i_2, \dots, i_μ , the individual constant 0, the accent ' as a notation for successor (of a number), the notation () for application of a singular 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 $S_0^a P$ to infer P ;

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

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

.....

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

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

.....

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

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

.....

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

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

.....

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

$$\chi_1(0, 0, \dots, 0, t + g + 1) \equiv \chi_1(0, 0, \dots, 0, t + g) \vee$$

$$Q_{100\dots 0}[\chi_1(0, 0, \dots, 0, t), \dots, \chi_N(0, 0, \dots, 0, t), \chi_1(0, 0, \dots, 1, t), \dots, \chi_N(M+1, M+1, \dots, M+1, t)]$$

$$\chi_2(0, 0, \dots, 0, t + g + 1) \equiv \chi_2(0, 0, \dots, 0, t + g) \vee$$

$$\bar{\chi}_1(0, 0, \dots, 0, t + g) Q_{200\dots 0}[\chi_1(0, 0, \dots, 0, t), \dots,$$

$$\chi_N(0, 0, \dots, 0, t), \chi_1(0, 0, \dots, 1, t), \dots, \dots,$$

$$\chi_N(M + 1, M + 1, \dots, M + 1, t)]$$

.....

$$\chi_N(M, M, \dots, M, t + g + 1) \equiv \chi_N(M, M, \dots, M, t + g) \vee$$

$$\bar{\chi}_1(M, M, \dots, M, t + g) \bar{\chi}_2(M, M, \dots, M, t + g) \dots$$

$$\bar{\chi}_{N-1}(M, M, \dots, M, t + g) Q_{NM\dots M}[\chi_1(0, 0, \dots,$$

$$0, t), \dots, \chi_N(0, 0, \dots, 0, t), \chi_1(0, 0, \dots, 1, t),$$

$$\dots, \dots, \chi_N(2M + 1, 2M + 1, \dots, 2M + 1, t)]$$

$$\chi_1(x_1 + M + 1, 0, \dots, 0, t + g + 1) \equiv \chi_1(x_1 + M + 1, 0,$$

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

$$\chi_2(x_1 + M + 1, 0, \dots, 0, t + g + 1) \equiv \chi_2(x_1 + M + 1, 0, \dots, 0,$$

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

.....

$$\chi_1(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g + 1) \equiv$$

$$\chi_1(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g) \vee$$

$$Q_1[\chi_1(x_1, x_2, \dots, x_m, t), \dots, \chi_N(x_1, x_2,$$

$$\dots, x_m, t), \chi_1(x_1, x_2, \dots, x_m + 1, t), \dots,$$

$$\dots, \chi_N(x_1 + 2M + 2, x_2 + 2M + 2, \dots,$$

$$x_m + 2M + 2, t)]$$

$$\chi_2(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g + 1) \equiv$$

$$\chi_2(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g) \vee$$

$$\bar{\chi}_1(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g) Q_2[\chi_1(x_1,$$

$$x_2, \dots, x_m, t), \dots, \chi_N(x_1, x_2, \dots, x_m, t),$$

$$\chi_1(x_1, x_2, \dots, x_m + 1, t), \dots, \dots,$$

$$\chi_N(x_1 + 2M + 2, x_2 + 2M + 2, \dots, x_m + 2M + 2, t)]$$

.....

$$\chi_N(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g + 1) \equiv$$

$$\chi_N(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g) \vee$$

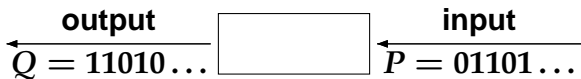
$$\bar{\chi}_1(x_1 + M + 1, x_2 + M + 1, \dots, x_m + M + 1, t + g) \bar{\chi}_2(x_1 + M + 1,$$

$$x_2 + M + 1, \dots, x_m + M + 1, t + g) \dots \bar{\chi}_{N-1}(x_1 + M + 1, x_2 + M + 1,$$

$$\dots, x_m + M + 1, t + g) Q_N[\chi_1(x_1, x_2, \dots, x_m, t), \dots,$$

$$\chi_N(x_1, x_2, \dots, x_m, t), \chi_1(x_1, x_2, \dots, x_m + 1, t), \dots,$$

Game-Theoretic View



For $t = 0, 1, 2, \dots$: 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.

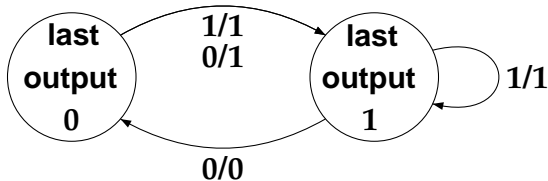
Play $\begin{pmatrix} P(0) \\ Q(0) \end{pmatrix} \begin{pmatrix} P(1) \\ Q(1) \end{pmatrix} \begin{pmatrix} P(2) \\ Q(2) \end{pmatrix} \dots$ is won by 2 if $(\mathbb{N}, \dots) \models \varphi(P, Q)$

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).

Church's Problem (1962)

- Given an MSO-formula $\varphi(X, Y)$ as requirement, decide whether it can be realized by a Mealy automaton, and if yes construct such an automaton.
- In other words:
Decide whether Player 2 has a winning strategy, and if this is the case construct a finite-state winning strategy.

Büchi-Landweber Theorem (1969)

For each MSO-requirement $\varphi(X, Y)$ either Player 1 or Player 2 has a finite-state winning strategy.

It is decidable who wins, and a finite-state winning strategy for the respective winner is computable.

Overview of Proof

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

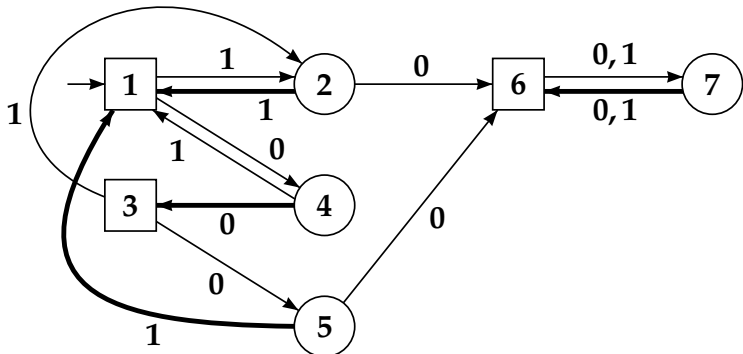
Steps 1 and 2 go from logic to automata (and games).

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

Steps 1 and 2: Example

$\varphi(X, Y)$:

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



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

- **Closer analysis (this talk)**
 1. Definability of controllers
 2. Generalizing winning strategies
 3. Memory-optimal controllers
 4. Optimal solutions for liveness requirements

Definability of Controllers

Strategies and Definability

A strategy for Player 1 is a map

$$\binom{P(0)}{Q(0)} \binom{P(1)}{Q(1)} \cdots \binom{P(k)}{Q(k)} \mapsto 0/1$$

A strategy for player 2 is a map

$$\binom{P(0)}{Q(0)} \binom{P(1)}{Q(1)} \cdots \binom{P(k)}{*} \mapsto 0/1$$

Logical definition by formula $\psi(X, Y, z)$

which is interpreted over finite play prefixes:

For Player 2: Pick bit 1 in round k iff

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

Definability Results

- Büchi-Elgot-Trakhtenbrot (1957-1960):
Finite-state strategies are MSO-definable.
- A. Rabinovich, W. Th. (CSL 2007):
First-order specifications can be solved by first-order definable strategies
Two versions: $FO(S)$, $FO(<)$
- In particular:
LTL-specification can be solved with LTL-definable controllers.

General Problem:

Relate the logic for describing the controller's behavior to the logic for describing the requirement.

Presburger Arithmetic

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

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

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

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

$$x^2 = y \Leftrightarrow$$

$$y \in \text{Squ} \wedge y - (2x - 1) \text{ is the greatest square } < y$$

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

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

Generalizing Winning Strategies

Sequence Transformations

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

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

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

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

is solvable only by the non-continuous operator T_0 with

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

Special Cases of 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

$$T^-: P(0)P(1) \dots \mapsto P(0)P(2)P(4) \dots$$

T^- is continuous, with linearly increasing delay.

Example 2: Doubling a sequence

$$T^+: P(0)P(1) \dots \mapsto P(0)P(0)P(1)P(1) \dots$$

T^+ is bit-by-bit-computable, but not finite-state-computable.

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
(Standard procedure derived from DFA minimization)
- View the game graph as an automaton and reduce it first
(Holtmann, Löding)
- Search the space of all (winning) strategies to find one
with minimal-memory implementation
(open, hint by Büchi-Landweber)

Optimal Solutions for Liveness Requirements

Optimality in Request-Response Games

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

Subsets $Rqu_1, \dots, Rqu_k \subseteq V$: “Requests”

Subsets $Rsp_1, \dots, Rsp_k \subseteq V$: “Responses”

RR-condition:

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

LTL:

$$\bigwedge_{i=1}^k \mathbf{G}(Rqu_i \rightarrow \mathbf{X}F Rsp_i)$$

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

- **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

Activation of i -th condition in a play q 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 both linear and quadratic penalty define:

- $w_\varrho(n) = \text{sum of penalties in } \varrho(0) \dots \varrho(n) \text{ divided by number of activations}$

”average penalty sum per activation”

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

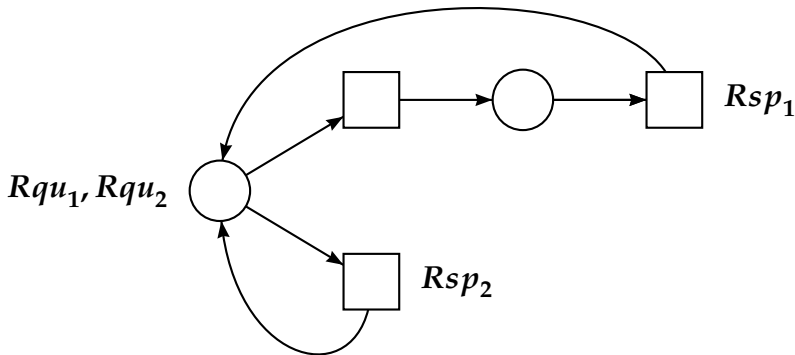
Given a strategy σ for controller and a strategy τ for adversary

- $\varrho(\sigma, \tau) := \text{the play induced by } \sigma \text{ and } \tau$
- $w(\sigma) := \sup_{\tau} w(\varrho(\sigma, \tau))$

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 the quadratic penalty function
(in fact, for any strictly increasing divergent penalty)
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, \dots, n_k)

$n_i =$

current waiting time for i -th condition since last activation

Derived mean-payoff game:

For each edge $e = (u, \bar{m}) \rightarrow (v, \bar{n})$

introduce edge weight

$w(e) = n_1 + \dots + n_k$ (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, \bar{m}, m) with $v \in V, \bar{m} \in [0, B]^{k-1}, m \geq 0$

In a play with unbounded waiting times for the last condition, pick a “critical segment” $(v, \bar{m}, m), \dots, (v, \bar{m}, m')$ 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)

Conclusion

- Church's Problem is far from closed.
- A current perspective is to turn decision problems into optimization problems.

Sources:

1. A. Rabinovich, W. Th., Logical refinements of Church's Problem, CSL 2007
2. M. Holtmann, C. Löding, Memory reduction for strategies in infinite games, CIAA 2007
3.
 - N. Wallmeier, PhD Thesis, RWTH Aachen 2007
 - F. Horn, W. Th., N. Wallmeier, Optimal strategy synthesis for request-response games (in preparation)