Facets of Synthesis: Revisiting Church's Problem

Wolfgang Thomas

RWTH Aachen University, Lehrstuhl Informatik 7, Aachen, Germany
thomas@informatik.rwth-aachen.de

Abstract. In this essay we discuss the origin, central results, and some perspectives of algorithmic synthesis of nonterminating reactive programs. We recall the fundamental questions raised more than 50 years ago in "Church's Synthesis Problem" that led to the foundation of the algorithmic theory of infinite games. We outline the methodology developed in more recent years for solving such games and address related automata theoretic problems that are still unresolved.

1 Prologue

The objective of "synthesis" is to pass from a specification to a program realizing it. This is a central task in computer science, and – unfortunately or fortunately, depending on the point of view – its solution cannot in general be automatized. However, there are classes of specifications for which algorithmic synthesis is possible. In the present paper we deal with a fundamental class of this kind, the regular specifications for nonterminating reactive programs. In another terminology, this amounts to the solution of infinite two-person games where the winning condition is given by a regular ω-language. The central problem on these games (Church's Problem [4,5]) and the first solutions (Büchi and Landweber [3], Rabin [22]) date back decades by now. Starting from this pioneering work, the algorithmic theory of infinite games grew into a very active area of research, and a fascinating landscape of models and algorithmic solutions is developing, covering, for example, timed games, weighted games, games over infinite state spaces, distributed games, stochastic games, concurrent games, and multiplayer games. Rather than trying an overview of these areas we return in this paper to the fundamentals. In an informal style addressing a general audience[1], we discuss three issues. First, we recall the problem, its connection with infinite games, and explain some of its historical context in mathematics (more precisely, set theory). Then the main techniques for solving infinite games are outlined. Finally, we discuss some questions around the concept of uniformization that are directly connected with the original problem and seem worth further study.

[1] We assume, however, that the reader knows the basic concepts about automata and logics over infinite words, like Büchi automata and monadic second-order logic over the structure $(\mathbb{N}, +1)$; see [13] as a reference.

2 Church's Problem

An instance of the synthesis problem is the description of a desired relation $R(X, Y)$ between inputs X (say, from a domain D) and outputs Y (say, from a domain D'). From the description of R we want to construct a program $\mathcal{P}$, computing a function $f_\mathcal{P} : D \to D'$ such that

$$\forall X \in D \; R(X, f_\mathcal{P}(X))$$

At the "Summer Institute of Symbolic Logic" at Cornell in 1957, Alonzo Church posed the following problem in [4] (see also [5]):

> *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).*

Church considers the case that the input X is an infinite sequence $X_0X_1\ldots$ of data (of a finite domain, we consider just bits in this paper) from which *in an on-line mode* the output sequence $Y = Y_0Y_1\ldots$ (again consisting of bits) has to be generated. The desired program is here a "circuit" (which amounts to a finite automaton); it has to produce the output bit Y_i from the input bits $X_0 \ldots X_i$.

The relation $R(X, Y)$ between inputs and outputs can be identified with an ω-language L_R. We associate with each pair $(X, Y) = (X_0X_1X_2\ldots, Y_0Y_1Y_2\ldots)$ of sequences the ω-word

$$X^\wedge Y := X_0Y_0X_1Y_1X_2Y_2\ldots$$

and let $L_R = \{X^\wedge Y | R(X, Y)\}$. A relation R is called *regular* if the associated ω-language L_R is regular, i.e. definable by a Büchi automaton or by a monadic second-order formula (MSO-formula) $\varphi(X, Y)$ over the structure $(\mathbb{N}, +1)$. As solutions we use finite automata with output, more precisely Mealy automata. Over a finite state-space S and with input alphabet Σ_1 and output alphabet Σ_2, a Mealy automaton is equipped with a transition function $\sigma : S \times \Sigma_1 \to S$ and an output function $\tau : S \times \Sigma_1 \to \Sigma_2$; in our case we set $\Sigma_1 = \Sigma_2 = \{0, 1\}$. Given such an automaton $\mathcal{A}$, the definition of the computed function $f_\mathcal{A} : \{0, 1\}^\omega \to \{0, 1\}^\omega$ is obvious.

Let us state Church's Problem in precise words for this case of MSO-definable relations and programs in the format of Mealy automata. (Other cases will be addressed later in this paper.)

> *Church's Problem:* Given an MSO-formula $\varphi(X, Y)$, defining the relation $R_\varphi \subseteq \{0, 1\}^\omega \times \{0, 1\}^\omega$, decide whether there is a Mealy automaton $\mathcal{A}$ such that $\forall X R_\varphi(X, f_\mathcal{A}(X))$ holds and – if yes – construct such a Mealy automaton from φ.

A dozen years later, Büchi and Landweber [3] solved this problem; Rabin [22] provided independently an alternative solution. The result established an ideal scenario within computer science: For the MSO-definable specifications, we can algorithmically check realizability, and we can in this case algorithmically construct a program (more precisely, an automaton) realizing the specification. In the context of construction of reactive systems, we view the Mealy automaton provided by a solution as a *finite-state controller* that behaves correctly for any behaviour of the environment as represented by input sequences. Since many interesting specifications are in fact regular, this result has great potential for practical applications. (One has to admit, however, that in order to realize this potential more work is necessary regarding the computational complexity of the algorithms involved.) Before we sketch the main ideas that enter the solution of Church's Problem, we introduce a simple connection of the problem with the theory of infinite games and underlying motivations from set theory.

3 Infinite Games, Determinacy, and Set Theory

A treatment of Church's Problem in the framework of infinite games was first proposed by McNaughton in an unpublished technical report [17], based on work of Gale and Stewart [12]. With each ω-language $L \subseteq \{0,1\}^\omega$ one associates an infinite game $G(L)$. (If L is regular, we say that the game $G(L)$ is regular.) It is played between two players. Due to the symmetry between them, we prefer to name them Player 1 and Player 2 rather than "Input" and "Output". The players choose bits in alternation (where Player 1 starts), forming a sequence $\varrho = X_0Y_0X_1Y_1X_2Y_2\ldots$ which is called "play" in this context. Player 2 wins the play ϱ if $\varrho \in L$, otherwise Player 1 wins ϱ. A *strategy* for Player 2 is a function $f : \{0,1\}^+ \to \{0,1\}$, mapping a finite sequence $X_0X_1\ldots X_i$ to the next chosen bit Y_i. A strategy f induces canonically a function $f_\omega : \{0,1\}^\omega \to \{0,1\}^\omega$ mapping a sequence $X_0X_1\ldots$ to a sequence $Y_0Y_1\ldots$ If for each sequence $X_0X_1\ldots$ the play $X_0Y_0X_1Y_1\ldots$ resulting from applying f belongs to L, f is called winning strategy for Player 2. So winning strategies capture the type of function that are asked for in Church's Problem (disregarding however the aspect of finite-state computability). Analogously, one introduces strategies and winning strategies g for Player 1. Now we have $g : Y_0\ldots Y_{i-1} \mapsto X_i$, so $g : \{0,1\}^* \to \{0,1\}$. We say that g is a winning strategy g for Player 1 if for each $Y_0Y_1\ldots$ the sequence $X_0Y_0X_1Y_1\ldots$ obtained by applying g does not belong to L.

By this formulation one introduces some symmetry into the model, which is not present in the formulation as given by Church. In the mathematical theory of infinite games this symmetry is a dominating aspect. The main question in the theory is the following problem: Given L, can we show that either Player 1 or Player 2 has a winning strategy? In this case one calls the game $G(L)$ *determined*. Determinacy is the dichotomy

$$\exists \text{ strategy } f\ \forall X \quad X^\wedge f_\omega(X) \in L \quad \vee \quad \exists \text{ strategy } g\ \forall Y \quad g_\omega(Y)^\wedge Y \notin L$$

Determinacy may at first seem a superfluous aspect in the study of Church's Problem. It looks useless to know that a specification – say modeling the relation between a program and its environment – can either be realized by the program or that the complement of it can be realized by the environment. However, there are several good reasons to address determinacy. In this section we discuss a first such motivation from set theory (and the reader not interested in historical connections can skip this discussion). Later we treat also other uses of determinacy.

Given an ω-language L, we consider a slightly modified game $G^*(L)$, since in this case the connection to set theory comes in very naturally. Here Player 1 picks *finite sequences* of bits (rather than single bits) when it is his turn, whereas Player 2 stays with picking single bits. We write $G^{**}(L)$ if both players pick finite sequences of bits in each move (this is the so-called Banach-Mazur game).

Let us focus on $G^*(L)$. Intuition tells us that for a "very large" set L it will be much easier to guarantee a winning strategy for Player 2 than for a "very small" set L. Now the question of comparing sizes of infinite sets is at the heart of Cantor's set theory. In fact, Cantor conjectured in his "Continuum Hypothesis" CH a dichotomy: that each set $L \subseteq \{0,1\}^\omega$ is either "very small" (and he meant "being at most countable" by this) or "very large" (meaning that its cardinality is the same as that of the full space $\{0,1\}^\omega$). Often, CH is formulated as the claim that between the cardinalities $|\mathbb{N}|$ and $|\mathbb{R}|$ there are no further cardinalities; since $|\{0,1\}^\omega| = |\mathbb{R}|$, this claim is equivalent to the one formulated above.

Cantor could not show this dichotomy – and today we know that CH is indeed independent of the standard set theoretical assumptions as formulated in the axiom system ZFC. However, the so-called Cantor-Bendixson Theorem shows that CH is true at least for all "closed" sets L. A set $L \subseteq \{0,1\}^\omega$ is *closed* if one can infer $X \in L$ from the condition that infinitely many finite prefixes w of X have an extension wZ in L. One calls L *open* if its complement $\overline{L}$ is closed.

We now state the Cantor-Bendixson Theorem and see that it amounts to a game-theoretic dichotomy, namely that for an open set L the game $G^*(L)$ is determined. The statement of the theorem refers to a geometric view of the space $\{0,1\}^\omega$, in the sense that we consider $\{0,1\}^\omega$ as the set of all infinite paths through the binary tree T_2 (where 0 means "branch left" and 1 means "branch right").

A pattern of paths as indicated in Figure 1 below we call *tree copy*; it is given by infinitely many branching points (bullets in the figure), such that from each of the two sons (circles in the figure) of a branching point, a finite path to a next branching point exists. We say that a set $L \subseteq \{0,1\}^\omega$ *allows a tree copy* if

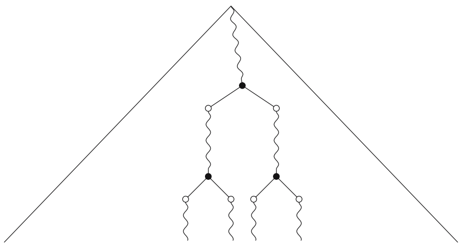

Fig. 1. A tree copy

there is a tree copy such that each path passing through infinitely many of its branching points belongs to L. Obviously a set L that allows a tree copy must have the same cardinality as $\{0,1\}^\omega$. Using this simplifying terminology (instead of the official one involving "perfect sets"), the Cantor-Bendixson Theorem says: *Each closed set L is either at most countable or allows a tree copy.*

Let us infer that for open K, the game $G^*(K)$ is determined. By the Cantor-Bendixson Theorem we know that $L := \overline{K}$ is either countable or allows a tree copy. In the *first case*, Player 2 can apply a diagonalization method: He uses an enumeration $Z_0, Z_1, \ldots$ of L and chooses his i-th bit in a way to make the resulting play different from Z_i. Then the play will be outside L and hence in K, so this is a winning strategy for Player 2 in $G^*(K)$. In the *second case*, Player 1 refers to the tree copy of L and chooses his bit sequences in a way to remain inside this tree copy (he always moves "to the next branching point"). Then the play will be in L and hence outside K, so this is a winning strategy for Player 1 in $G^*(K)$.

For the games $G(L)$, determinacy is not so easily connected with cardinalities of sets. Nevertheless, topological notions (such as "open" and "closed") are the key for showing determinacy results. The fundamental result, due to Martin, rests on the notion of Borel set: An ω-language is *Borel* if it can be built from open and closed sets in an at most countable (possibly transfinite) sequence of steps, where each single step consists of a countable union or a countable intersection of already constructed sets. Martin's Theorem now says that for a Borel set L, the game $G(L)$ is determined (see e.g. [16]). As will be explained later, all regular games are Borel and hence determined.

For the Banach-Mazur games $G^{**}(L)$, nice connections can again be drawn to concepts of richness of sets (like "co-meager"); a recent in-depth analysis of the determinacy problem is Grädel's work [11].

4 Solving Infinite Games

In this section we give a very brief sketch of the techniques that enter (known) solutions of Church's Problem, using game-theoretic terminology. For a detailed development see e.g. the tutorial [25].

4.1 From Logic to Games on Graphs

The start is a conversion of the originally logical problem into an automata theoretic one, by a transformation of the given MSO-formula φ into an equivalent ω-automaton. Here we apply the well-known results due to Büchi [2] and McNaughton [18] that allow to convert an MSO-formula into an equivalent (nondeterministic) Büchi automaton, and then a Büchi automaton into an equivalent (deterministic) Muller automaton. [2] A Büchi automaton has a designated set F of final states; and a run is accepting if some state of F occurs infinitely often in

[2] It should be noted that in the early days of automata theory, the conversion of regular expressions or logical formulas into automata was called "synthesis" and the converse "analysis" (see e.g. the introction of [1]).

it. The run of a Muller automaton with finite state set Q is accepting if the set of states visited infinitely often in it belongs to a predefined collection $\mathcal{F} \subseteq 2^Q$ of state sets.

We can easily build this automaton in a way that the processing of letters X_i contributed by Player 1 is separated from the processing of letters Y_i contributed by Player 2, in the sense that the state set Q is partitioned into two sets Q_1 (from where bits of Player 1 are read) and Q_2 (from where bits of Player 2 are read). Thus a run of the automaton switches back and forth between visits to Q_1- and to Q_2-states. Since the acceptance of a play by the automaton refers only to the visited states, we may drop the input- and output-letters for our further analysis and identify a play with a state sequence through the automaton. The resulting structure is also called *game arena* or *game graph*: We imagine that the two Players 1 and 2 build up a path in alternation – Player 1 picks a transition from a state in Q_1, similarly Player 2 from a state in Q_2. The winning condition (for Player 2) is now no more a logic formula but a very simple requirement, namely that the states visited infinitely often in a play form a set in the predefined collection $\mathcal{F}$. Referring to this "Muller winning condition", we speak of a *Muller game* over a finite graph G.

For solving Church's Problem, it is now sufficient to decide whether for plays beginning in the start state of the graph, Player 2 has a winning strategy, and in this case to construct such a strategy in the form of a Mealy automaton. Due to the deletion of the transition labels, the Mealy automaton now maps a finite play prefix from Q^* to a "next state" from Q. For a vertex v the play ϱ starting from v is won by Player 2 iff

$$\text{for some } F \in \mathcal{F} : \exists^\omega i \; \varrho(i) \in F \wedge \neg\exists^\omega i \; \varrho(i) \notin F$$

(here $\exists^\omega$ is the quantifier "there exist infinitely many"). From this form of the winning condition it is easy to see that the set L_v of plays won by Player 2 from v is Borel, and in fact a Boolean combination of countable intersections of open sets. Hence the Muller game over G with start vertex v is determined.

We add some methodological remarks.

1. The main effect of this transformation is the *radical simplification of the winning condition* from a possibly complex logical formula to the requirement that a play visits certain states infinitely often and others only finitely often. This simplification is made possible by distinguishing finitely many different "game positions" in the form of the automaton states. As mentioned, we can infer that a game as presented in Church's Problem is determined. The cost to be payed for this simplicity is the high number of states; it is known that in the length of (MSO-) formulas this number grows at a rate that cannot be bounded by an elementary function.
2. It should be noted that *all known solutions of Church's Problem involve this reduction from logic to automata (or graphs).* In model-checking, similar remarks apply when the linear-time logic LTL is considered; on the other hand, CTL-model-checking proceeds directly by an induction on formulas, which is one reason for its efficiency. It would be interesting to know logics of

substantial expressive power that allow a similar approach for the solution of games.

3. The introduction of game graphs has another advantage: In modelling reactive systems, it is usually convenient to describe the interaction between a controller and its environment by a game graph, adding a specification in the form of a winning condition (then as an ω-language over the respective state set Q). In practice this winning condition may not be a Muller condition but again an MSO-formula or LTL-formula φ, defining an ω-language $L \subset Q^\omega$. In this case one can also apply the above-mentioned transformation of φ into a Muller automaton (call its state set S), obtaining a game graph over the vertex set $S \times Q$, now together with a Muller condition (this condition just refers to the S-components of visited states).

4.2 Parity Games

The direct solution of Muller games (as presented in the difficult original paper [3]) is rather involved. It is helpful to pass to a different kind of game first, called *parity game*, and then solve this parity game. As for the Muller winning condition, also the parity winning condition is a Boolean combination of statements that certain states are visited infinitely often. But instead of a collection $\mathcal{F}$ of state sets, a uniform coloring of vertices by a finite list of colors is used. We take here natural numbers as colors. A play ϱ is won by Player 2 iff the highest color visited infinitely often during ϱ is even. This amounts to the disjunction over the following statements for all even $i \leq k$: Color i is visited infinitely often but each color $j > i$ only finitely often. This winning condition was proposed first by Mostowski [19], and it has a precursor in the "difference hierarchy" in Hausdorff's *Grundzüge der Mengenlehre* [14], introduced there to structure the levels of the Borel hierarchy (see also [24]).

Technically, one reduces Muller games to parity games in the following sense: For a game graph G and a collection $\mathcal{F}$, defining a Muller winning condition, one constructs a new graph G' with an appropriate coloring c such that each play ϱ in G induces a corresponding play ϱ' in G' with the following property: Player 2 wins ϱ under the Muller condition w.r.t. $\mathcal{F}$ iff Player 2 wins ϱ' under the parity condition w.r.t. c. In the standard construction of G', using the "latest appearance record", one introduces memory about the visited vertices of G in the order of their last visits. Thus a vertex of G' is basically a permutation of vertices in G (the leading entry being the current vertex of G); so the transformation $G \mapsto G'$ involves a serious blow-up. However, the solution of parity games is much easier than that of Muller games. In particular, *memoryless winning strategies* suffice for the respective winner; i.e. strategies that do not involve memory on the past of a play but just depend on the respective currently visited vertex.

As a preparation for the solution of parity games, one considers a very simple winning condition, called *reachability condition.* We are given a set F of "target vertices", and Player 2 wins the play ϱ if a vertex from F occurs in ϱ. To solve a reachability game means to compute those vertices v from which Player 2 has a strategy to force a visit in F.

This problem has a straightforward solution: Over the (finite) vertex set Q, one computes, for $i = 0, 1, 2, \ldots$, the set $A_i^2(F)$ of those states from which Player 2 can guarantee a visit in F within i steps. Obviously $A_0^2(F) = F$, and we have $v \in A_{i+1}^2(F)$ iff one of the two following cases holds:

– $v \in Q_2$ and one edge connects v with a vertex in $A_i^2(F)$,
– $v \in Q_1$ and each edge from v leads to a vertex in $A_i^2(F)$.

We have $A_0^2(F) \subseteq A_1^2(F) \subseteq A_2^2(F) \ldots$ and set $A^2(F) = \bigcup_i A_i^2(F)$; this set is obtained at some stage i_0. Clearly from each vertex in $A^2(F)$, Player 2 has a winning strategy by taking edges which decrease the distance to F with each step (and it is also easy to see that a memoryless strategy suffices). To show that the construction is complete, we verify that from each vertex outside $A^2(F)$ the other player (1) has a winning strategy. Thus we show that reachability games are *determined*. This use of determinacy is a general method: To show completeness of a game solution one verifies a determinacy claim.

How is this done for a reachability game? From the construction of the $A_i^2(F)$ it is clear that for each v in the complement of $A^2(F)$ one of the following cases holds:

– $v \in Q_2 \setminus A^2(F)$ and hence no edge leads from v to some $A_i^2(F)$; so each edge from v leads to a vertex again outside $A^2(F)$,
– $v \in Q_1 \setminus A^2(F)$ and hence not each edge leads from v to some $A_i^2(F)$; so some edge from v leads to a vertex outside $A^2(F)$.

Clearly this yields a strategy for Player 1 to avoid visiting $A^2(F)$ from outside $A^2(F)$, and hence to avoid visiting F; so we have a winning strategy for Player 1.

The set $A^2(F)$ as constructed above for the set F is often called the "attractor of F (for Player 2)". As it turns out, an intelligent iteration of this construction basically suffices also for the solution of parity games. The determinacy claim even holds without the assumption that the game graph is finite: *In a parity game over the game graph G, for each vertex v one of the two players has a memoryless winning strategy for the plays starting from v. If G is finite, one can decide who wins and compute a memoryless winning strategy* ([7]).

The idea to show this is by induction on the number of used colors, and the rough course of the induction step is as follows: Let Q be the set of vertices and W_1 be the set of vertices from which Player 1 has a memoryless winning strategy. We have to show that from each vertex in $Q \setminus W_1$, Player 2 has a memoryless winning strategy. Consider the set C_k of vertices having the maximal color k. We assume that k is even (otherwise one has to switch the players). The case $C_k \subseteq W_1$ gives us a game graph $Q \setminus W_1$ without color k; applying the induction hypothesis to this set we easily get the claim. Otherwise consider $C_k \setminus W_1$ and define its attractor A for Player 2, restricted to $Q \setminus W_1$. (For the case of an infinite game graph, the inductive definition of $A^2(F)$ mentioned above has to be adapted, involving a transfinite induction.) Now the vertex set $Q \setminus (W_1 \cup A)$ defines a game graph without color k, so by induction hypothesis there exists a division into two sets U_1, U_2 from where Player 1, respectively Player 2 has a memoryless winning strategy. It turns out that U_1 must be empty and that on $A \cup U_2$, which is the complement of W_1, Player 2 has a memoryless winning strategy.

Over finite game graphs, also the effectiveness claims can be shown; it is open whether the decision who wins a parity game from a given vertex v is possible in polynomial time.

5 Model-Checking and Tree Automata Theory

We have mentioned two motivations for the question of determinacy, a historical one in set theory, and a technical one regarding the completeness of winning strategies. In this section we address a third type of motivation, originating in logic. In this case determinacy reflects the duality between "true" and "false". We indicate very briefly two kinds of application. Details can be found, e.g., in [13].

5.1 Model-Checking

Model-Checking is the task of evaluating a formula φ in a structure $\mathcal{S}$. It is well-known that the evaluation can be explained in terms of a finite game between the players Proponent (aiming at showing truth of the formula) and Opponent (aiming at the converse). The duality between the connectives $\vee, \wedge$ and between the quantifiers $\exists, \forall$ is reflected in the rules: For example, when treating $\varphi_1 \vee \varphi_2$ then Proponent picks one of φ_1, φ_2 for which the evaluation continues, and when a formula $\varphi_1 \wedge \varphi_2$ is treated, then Opponent picks one of φ_1, φ_2. Proponent wins a play if it ends at an atomic formula which is true in $\mathcal{S}$. Then $\mathcal{S}$ satisfies φ iff in this finite "model-checking game" Proponent has a winning strategy (and falsehood means that Opponent has a winning strategy).

The modal μ-calculus is a logic involving least and greatest fixed points rather than quantifiers; the formulas are interpreted in transition graphs in the form of Kripke structures. Even for finite Kripke structures, the appropriate model-checking game turns out to be infinite, since the semantics depends on infinite paths when fixed point operators enter. As first shown in [8], the model-checking game for the μ-calculus is in fact a parity game; the game graph is a product of the considered structure $\mathcal{S}$ and the collection $\mathrm{SF}(\varphi)$ of subformulas of the given formula φ. Thus the open problem about polynomial time solvability of the μ-calculus model-checking poblem reduces to the (again open) problem of polynomial time solvability of parity games.

An extension of the μ-calculus model-checking game to a "quantitative μ-calculus" was developed in [9].

5.2 Tree Automata

A very strong decidability result of logic is Rabin's Tree Theorem, saying that the MSO-theory of the infinite binary tree is decidable [21]. Rabin's method for showing this was a transformation of MSO-formulas (interpreted over the binary tree T_2) into finite tree automata working over labelled infinite trees (where each node of T_2 gets a label from a finite alphabet Σ). These tree automata are nondeterministic, and a run is a labelling of the input tree nodes with automaton states such that a natural compatibility with the input tree and the automaton's transition relation holds. As acceptance condition for a given run one can require that on each path of the run, a parity condition is satisfied. In this case we speak of a parity tree automaton.

The main problem of this approach is to show complementation for parity tree automata. For this the determinacy of parity games (in the version over infinite game graphs) can be used. In fact, the acceptance of a labelled tree t by a parity tree automaton $\mathcal{A}$ can be captured in terms of a parity game $G_{\mathcal{A},t}$ between two players called "Automaton" and "Pathfinder". The infinite game graph is a kind of product of t and $\mathcal{A}$. Then $\mathcal{A}$ accepts t iff Automaton has a winning strategy in $G_{\mathcal{A},t}$. The complementation proof (following [10]) starts with the negation of the statement "$\mathcal{A}$ accepts t", i.e. in game-theoretical terms with the statement "Automaton does not have a (memoryless) winning strategy in $G_{\mathcal{A},t}$". By memoryless determinacy of parity games, this means that Pathfinder has a memoryless winning strategy in $G_{\mathcal{A},t}$. From this winning strategy it is possible to build (in fact, independently of the given tree t) a new tree automaton $\mathcal{B}$ as the complement automaton for $\mathcal{A}$.

There are further results that belong to the very tight connection between infinite games and automata on infinite trees. In particular, the solution of parity games over finite graphs is reducible to the emptiness problem for parity tree automata and conversely.

Let us comment on the application of tree automata for the solution of games. This use of tree automata appears in Rabin's approach to Church's Problem (developed in [22]). The idea is to code a strategy of Player 2 by a labelling of the nodes of the infinite binary tree: The root has no label, the directions left and right represent the bits chosen by Player 1, and the labels on the nodes different from the root are the bits chosen by Player 2 according to the considered strategy. When Player 1 chooses the bits $b_0, \ldots, b_k$, he defines a path to a certain node; the label b of this node is then the next choice of Player 2. Now the paths through a labelled tree t capture all plays that are compatible with Player 2's strategy coded by t. Using analogous constructions to those explained in Section 3 above, one can build a parity tree automaton $\mathcal{A}$ that checks whether t codes a winning strategy. Deciding non-emptiness of $\mathcal{A}$ thus allows to decide whether Player 2 wins the given game. By Rabin's "Basis Theorem" we even know that in this case some regular tree is accepted by $\mathcal{A}$. This regular tree can then be interpreted as a finite-state winning strategy for Player 2.

6 On Uniformization

A class $\mathcal{R}$ of binary relations $R \subseteq D \times D'$ is *uniformizable* by functions in a class $\mathcal{F}$ if for each $R \in \mathcal{R}$ there is a function $f \in \mathcal{F}$ such that

- the graph of f is contained in R,
- the domains of R and f coincide.

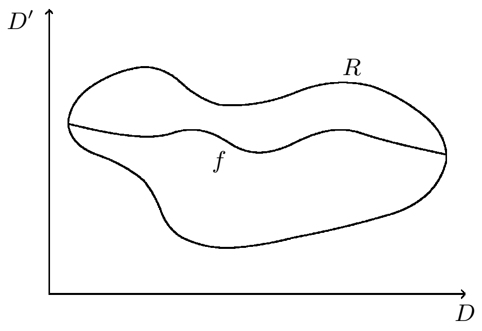

Fig. 2. Uniformization

Two well-known examples from recursion theory and from automata theory are concerned with the recursively enumerable, respectively the rational relations; here we have the "ideal" case that the graphs of the required functions are precisely of the type of the given relations.

Recall that a partial function from $\mathbb{N}$ to $\mathbb{N}$ is recursive iff its graph is recursively enumerable. The Uniformization Theorem of recursion theory says that a binary recursively enumerable relation R is uniformizable by a function whose graph is again recursively enumerable, i.e. by a (partial) recursive function f. A computation of $f(x)$ works as follows: Enumerate R until a pair (y, z) is reached with $x = y$, and in this case produce z as output.

A binary rational relation is defined (for instance) by a finite nondeterministic two-tape automaton that scans a given word pair (u, v) asynchronously, i.e. with two reading heads that move independently from left to right over u, respectively v. Rational relations are uniformizable by rational functions, defined as the functions whose graph is a rational relation (see e.g. [6]).

Church's Problem is a variant of the uniformization problem. It asks for the *decision* whether an MSO-definable relation is uniformizable by a Mealy automaton computable function. In the context of determinacy, the problem is extended to the question whether for an MSO-definable relation $R \subseteq \{0,1\}^\omega \times \{0,1\}^\omega$, either R itself or the complement of its inverse is uniformizable by a Mealy computable function.

There are numerous variants of this problem, either in the unilateral version (as in Church's Problem) or in the determinacy version. For both parameters, the class $\mathcal{R}$ of relations and the class $\mathcal{F}$ of uniformizing functions, there are several other interesting options.[3]

Let us first mention some natural classes of *functions*. A Mealy automaton computable function $f : \{0,1\}^\omega \to \{0,1\}^\omega$ is *continuous* (in Cantor's topology): Each bit of $f(X)$ only depends on a finite prefix of X. If the i-th bit of $f(X)$ only depends on $X_0, \ldots, X_i$ then we call f *causal*. One can show that a function is Mealy automaton computable iff it is causal and its graph is MSO-definable.

[3] In the monograph [26] of Trakhtenbrot and Barzdin one finds an early account of these matters.

The proof rests on the equivalence between MSO-logic and finite automata over *finite* words (the Büchi-Elgot-Trakhtenbrot-Theorem). Note that there are MSO-definable functions that are not causal and not even continuous (a trivial example is the function that maps a bit-sequence with infinitely many 1's to 1^ω and all other sequences to 0^ω).

A more restricted concept of MSO-definability refers to the fact that a strategy $f : \{0,1\}^+ \to \{0,1\}$ can be captured by the language

$$L_f := \{X_0 \dots X_i \in \{0,1\}^+ \mid f(X_0 \dots X_i) = 1\}$$

of finite words. We say that a *strategy is MSO-definable* if L_f is. For MSO-logic this is compatible with the definition given above: A strategy f is MSO-definable iff the graph of the associated causal function f_ω is MSO-definable.

A slightly more extended class of functions (over the causal ones) is given by the condition that the i-th bit of $f(X)$ only depends on the bits $X_0, \dots, X_j$ where $i < j$ and $j - i \leq k$ for some constant k; we then say that f is of *delay* k. In game-theoretic terms, this means that Player 2 can lag behind Player 1 by k moves. The dual type of function g is called "shift k" (where the i-th bit of $g(Y)$ depends on $Y_0, \dots, Y_j$ with $j < i$ and $i - j < k$). It is not difficult to show a determinacy result for MSO-definable relations by finite-state computable functions of delay k, respectively shift k. Hosch and Landweber [15] proved the interesting result that it is decidable whether for an MSO-definable relation the associated game is won by Player 2 with a strategy of bounded delay (by some k), and they also showed that in this case a finite-state winning strategy can be constructed.

Not much is known about more general strategies. One natural option is to consider strategies that induce functions $f : \{0,1\}^\omega \to \{0,1\}^\omega$ with linearly increasing delay, respectively linearly increasing shift. A function of the first type is the "division by 2" of sequences, mapping $X_0X_1X_2X_3X_4\dots$ to $X_0X_2X_4\dots$; an example of the second type is the "double function" $X_0X_1\dots \mapsto X_0X_0X_1X_1\dots$. Note that gsm-mappings (functions computable by generalized sequential machines) are finite-state computable functions of linearly increasing shift. These functions seem interesting for uniformization or determinacy problems on more general relations than the MSO-definable ones.

Regarding Church's Problem for other classes of *relations*, let us first consider relations that are first-order definable rather than MSO-definable. There are two natural versions of first-order logic, denoted FO(+1) and FO(<), where in brackets we exhibit the available arithmetical signature. (A technical point to be mentioned is that we cannot pass from a relation R to the associated ω-language L_R as explained in Section 2, since the even and the odd positions of an ω-sequence cannot be distinguished in FO(+1) and FO(<). So we consider a pair (X, Y) of bit sequences as an ω-word $(X_0, Y_0), (X_1, Y_1), \dots$ over the alphabet $\{0,1\} \times \{0,1\}$.) In [23], it was shown that a determinacy theorem holds for the FO(+1)-, respectively the FO(<)-definable relations, and that appropriate winning strategies exist which are again FO(+1)-, respectively FO(<)-definable.

There are also interesting logics for which the analogous result fails. We give an example for the unilateral case as addressed in Church's Problem. Consider

Presburger arithmetic, the first-order theory of addition over $\mathbb{N}$. We present a formula $\varphi(X, Y)$ of Presburger arithmetic such that in the game associated with R_φ there is a winning strategy for Player 2, however not a Presburger definable one. First we write down in Presburger arithmetic a formula $\varphi_{\text{squ}}(X)$ which says that X is the set Squ of squares (use the fact that the distances of successive squares increase by 2). Now one invokes the fact that multiplication is FO-definable in $(\mathbb{N}, +, \text{Squ})$ ([20]). Hence also each arithmetical set is FO-definable in $(\mathbb{N}, +, \text{Squ})$; pick such a set M which is not recursive, and let $\varphi_M(y)$ its FO-definition in $(\mathbb{N}, +, \text{Squ})$. Write $\varphi_M(X, y)$ for the formula where the predicate symbol for Squ is replaced by X. Now consider the following Presburger formula over $(\mathbb{N}, +)$:

$$\varphi(X, Y) := (\varphi_{\text{squ}}(X) \to \forall y(Y(y) \leftrightarrow \varphi_M(X, y)))$$

Clearly there is a winning strategy for Player 2 in the game defined by φ, e.g. with $Y = M$ for arbitrary X. As the case $X = \text{Squ}$ shows, the strategy f cannot be recursive. Thus the language L_f coding f is not recursive, so it cannot be Presburger definable.

It seems to this author that a comprehensive theory of effective determinacy and uniformization over infinite words is only in its beginnings. Although this question is raised from a theoretical point of view, results obtained in this research are likely to be interesting also in the context of the synthesis of controllers or reactive programs.

References

1. Büchi, J.R.: Weak second-order arihmetic and finite automata. Z. Math. Logik Grundlagen Math. 6, 66–92 (1960)
2. Büchi, J.R.: On a decision method in restricted second order arithmetic. In: Nagel, E., et al. (eds.) Proc. 1960 International Congress on Logic, Methodology and Philosophy of Science, pp. 1–11. Stanford University Press (1962)
3. Büchi, J.R., Landweber, L.H.: Solving sequential conditions by finite-state strategies. Trans. Amer. Math. Soc. 138, 367–378 (1969)
4. Church, A.: Applications of recursive arithmetic to the problem of circuit synthesis. In: Summaries of the Summer Institute of Symbolic Logic, vol. I, pp. 3–50. Cornell Univ., Ithaca (1957)
5. Church, A.: Logic, arithmetic, and automata. In: Proc. Int. Congr. Math. 1962, Inst. Mittag-Leffler, Djursholm, Sweden, pp. 23–35 (1963)
6. Eilenberg, S.: Automata, Languages, and Machines, vol. A. Academic Press, New York (1974)
7. Emerson, E.A., Jutla, C.S.: Tree automata, mu-calculus, and determinacy. In: Proc. 32nd FoCS 1991, pp. 368–377. IEEE Comp. Soc. Press, Los Alamitos (1991)
8. Emerson, E.A., Jutla, C.S., Sistla, A.P.: On model checking for fragments of the μ-calculus. In: Courcoubetis, C. (ed.) CAV 1993. LNCS, vol. 697, pp. 385–396. Springer, Heidelberg (1993)
9. Fischer, D., Grädel, E., Kaiser, L.: Model checking games for the quantitative μ-Calculus. In: Albers, S., Weil, P. (eds.) Proc. STACS 2008, pp. 301–312 (2008)
10. Gurevich, Y., Harrington, L.: Trees, automata, and games. In: Proc. 14th ACM Symp. on the Theory of Computing, pp. 60–65. ACM Press, New York (1982)

11. Grädel, E.: Banach-Mazur games on graphs. In: Proc. FSTTCS 2008 (2008), `http://drops.dagstuhl.de/portals/FSTTCS08`
12. Gale, D., Stewart, F.M.: Infinite games with perfect information. Ann. Math. Studies 28, 245–266 (1953)
13. Grädel, E., Thomas, W., Wilke, T. (eds.): Automata, Logics, and Infinite Games. LNCS, vol. 2500. Springer, Heidelberg (2002)
14. Hausdorff, F.: Grundzüge der Mengenlehre, Leipzig (1914)
15. Hosch, F.A., Landweber, L.H.: Finite delay solutions for sequential conditions. In: Nivat, M. (ed.) Proc. ICALP 1972, pp. 45–60. North-Holland, Amsterdam (1972)
16. Kechris, A.S.: Classical Descriptive Set Theory. Springer, New York (1995)
17. McNaughton, R.: Finite-state infinite games, Project MAC Rep., MIT, Cambridge, Mass (September 1965)
18. McNaughton, R.: Testing and generating infinite sequences by a finite automaton. Inf. Contr. 9, 521–530 (1966)
19. Mostowski, A.W.: Regular expressions for infinite trees and a standard form of automata. In: Skowron, A. (ed.) SCT 1984. LNCS, vol. 208, pp. 157–168. Springer, Heidelberg (1984)
20. Putnam, H.: Decidability and essential undecidability. JSL 22, 39–54 (1957)
21. Rabin, M.O.: Decidability of second-order theories and automata on infinite trees. Trans. Amer. Math. Soc. 141, 1–35 (1969)
22. Rabin, M.O.: Automata on infinite objects and Church's Problem. Amer. Math. Soc., Providence (1972)
23. Rabinovich, A., Thomas, W.: Logical Refinements of Church's Problem. In: Duparc, J., Henzinger, T.A. (eds.) CSL 2007. LNCS, vol. 4646, pp. 69–83. Springer, Heidelberg (2007)
24. Selivanov, V.: Fine hierarchies and Boolean terms. J. Smb. Logic 60, 289–317 (1995)
25. Thomas, W.: Solution of Church's Problem: A tutorial. In: Apt, K., van Rooij, R. (eds.) New Perspectives on Games and Interaction. Texts on Logic and Games, vol. 5, pp. 211–236. Amsterdam Univ. Press
26. Trakhtenbrot, B.A., Barzdin, Y.M.: Finite Automata. Behavior and Synthesis. North-Holland, Amsterdam (1973)