The Synthesis Problem

Angelo Montanari
Dept. of Mathematics, Computer Science, and Physics
University of Udine, Italy

Automatic System Verification: Theory and Applications

1. THE SYNTHESIS PROBLEM
   - Introduction to the synthesis problem
   - The solution schema

Warnings

As a matter of fact, the expression "synthesis problem" has been used to designate quite different problems.

WARNINGS

As a matter of fact, the expression "synthesis problem" has been used to designate quite different problems.

As an example, sometimes the synthesis problem is identified with the model building problem, that is, the problem of providing an algorithm that, given a formula of the logic under consideration, produces a complete description of a specific model of it, whenever it is satisfiable.
See, for instance, *French, McCabe-Dansted, and Reynolds, Synthesis for continuous time, Theoretical Computer Science, 594: 201-222, 2015*.

WARNINGS

As a matter of fact, the expression "synthesis problem" has been used to designate quite different problems.

As an example, sometimes the synthesis problem is identified with the model building problem, that is, the problem of providing an algorithm that, given a formula of the logic under consideration, produces a complete description of a specific model of it, whenever it is satisfiable.
See, for instance, *French, McCabe-Dansted, and Reynolds, Synthesis for continuous time, Theoretical Computer Science, 594: 201-222, 2015*.

Here, we refer to the original formulation of the synthesis problem given by Church and to the solution provided by Büchi and Landweber.

WARNINGS

As a matter of fact, the expression "synthesis problem" has been used to designate quite different problems.

As an example, sometimes the synthesis problem is identified with the model building problem, that is, the problem of providing an algorithm that, given a formula of the logic under consideration, produces a complete description of a specific model of it, whenever it is satisfiable.
See, for instance, *French, McCabe-Dansted, and Reynolds, Synthesis for continuous time, Theoretical Computer Science, 594: 201-222, 2015*.

Here, we refer to the original formulation of the synthesis problem given by Church and to the solution provided by Büchi and Landweber.

Our presentation of the problem and of the solution follow the tutorial: "Solution of Church's Problem: A Tutorial", by Wolfgang Thomas.

CHURCH'S PROBLEM

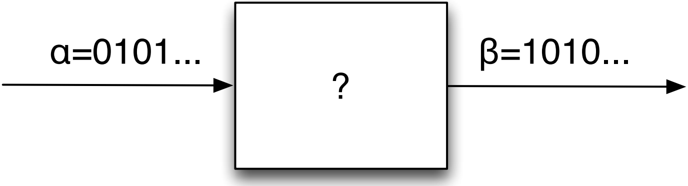
- The synthesis problem was originally proposed by Church during the "Summer Institute of Symbolic Logic" on 1957.

CHURCH'S PROBLEM

- The synthesis problem was originally proposed by Church during the "Summer Institute of Symbolic Logic" on 1957.
- It consists of the synthesis of a finite state machine (a circuit) which realizes a bit-to-bit transformation of an infinite sequence α into a corresponding infinite sequence β so that the pair (α, β) satisfies a specification expressed in a suitable (temporal) logic.

CHURCH'S PROBLEM

- The synthesis problem was originally proposed by Church during the "Summer Institute of Symbolic Logic" on 1957.
- It consists of the synthesis of a finite state machine (a circuit) which realizes a bit-to-bit transformation of an infinite sequence α into a corresponding infinite sequence β so that the pair (α, β) satisfies a specification expressed in a suitable (temporal) logic.
- Goal: given a specification of the input-output relation between α e β, build a corresponding machine:



THE RESULT BY BÜCHI AND LANDWEBER

- The problem consists in properly filling in the black box on the basis of the specification of the relationships between the input α and the output β.

THE RESULT BY BÜCHI AND LANDWEBER

- The problem consists in properly filling in the black box on the basis of the specification of the relationships between the input α and the output β.
- With respect to traditional (terminating) data manipulation programs, the focus switches from data with an infinite domain, which, in general, makes the synthesis problem undecidable, to infinite time.

THE RESULT BY BÜCHI AND LANDWEBER

- The problem consists in properly filling in the black box on the basis of the specification of the relationships between the input α and the output β.
- With respect to traditional (terminating) data manipulation programs, the focus switches from data with an infinite domain, which, in general, makes the synthesis problem undecidable, to infinite time.
- Surprisingly, Büchi and Landweber have shown that Church's problem admits a positive solution, that is, it is decidable, provided that the specification language (the temporal logic) is not too expressive.

AN EXAMPLE

Example

- $\forall t(\alpha(t) = 1 \rightarrow \beta(t) = 1)$
  - if, at a given time t, the input is 1, then the output is 1 as well;

AN EXAMPLE

Example

- $\forall t(\alpha(t) = 1 \to \beta(t) = 1)$
  - if, at a given time t, the input is 1, then the output is 1 as well;
- $\neg \exists t\ \beta(t) = \beta(t+1) = 0$
  - there are no two consecutive occurrences of 0 in the output sequence;

AN EXAMPLE

Example

- $\forall t(\alpha(t) = 1 \to \beta(t) = 1)$
  - if, at a given time t, the input is 1, then the output is 1 as well;
- $\neg\exists t\ \beta(t) = \beta(t+1) = 0$
  - there are no two consecutive occurrences of 0 in the output sequence;
- $\exists^{\omega} t\ \alpha(t) = 0 \to \exists^{\omega} t\ \beta(t) = 0$
  - if the input sequence features an infinite number of occurrences of 0, then the output sequence features an infinite number of occurrences of 0 as well.

AN EXAMPLE

Example

- $\forall t(\alpha(t) = 1 \to \beta(t) = 1)$
  - if, at a given time t, the input is 1, then the output is 1 as well;
- $\neg \exists t\ \beta(t) = \beta(t+1) = 0$
  - there are no two consecutive occurrences of 0 in the output sequence;
- $\exists^{\omega} t\ \alpha(t) = 0 \to \exists^{\omega} t\ \beta(t) = 0$
  - if the input sequence features an infinite number of occurrences of 0, then the output sequence features an infinite number of occurrences of 0 as well.

A solution procedure:

- if the input is 1, it produces the output 1;
- if the input is 0, it produces the output 1 if the previous output, on the input 0, was 0; otherwise, it produces the output 0.

FORMALIZATION OF THE PROBLEM

CONDITIONS ON TRANSFORMATIONS

The transformation we are looking for must satisfy the following conditions:

FORMALIZATION OF THE PROBLEM

CONDITIONS ON TRANSFORMATIONS

The transformation we are looking for must satisfy the following conditions:

1. bit-to-bit - when the machine receives the n-th character of α (as input), it must immediately produce the n-th character of β (as output). It follows that $\beta(n)$ may only depend on $\alpha(1), \ldots,$ $\alpha(n-1), \alpha(n)$;

FORMALIZATION OF THE PROBLEM

CONDITIONS ON TRANSFORMATIONS

The transformation we are looking for must satisfy the following conditions:

1. bit-to-bit - when the machine receives the n-th character of α (as input), it must immediately produce the n-th character of β (as output). It follows that $\beta(n)$ may only depend on $\alpha(1), \ldots,$ $\alpha(n-1), \alpha(n)$;
2. a finite state solution (machine) - to compute the output of a generic computation step (the output at time t), the machine needs to exploit a finite memory of a given size.

FORMALIZATION OF THE PROBLEM (CONT'D)

EXAMPLES

- $\beta(t) = \alpha(2t)$ (β returns the elements at even positions of α) violates condition 1 – the output symbol $\beta(t)$ must be produced without delay after receipt of the input symbol $\alpha(t)$;

FORMALIZATION OF THE PROBLEM (CONT'D)

EXAMPLES

- $\beta(t) = \alpha(2t)$ (β returns the elements at even positions of α) violates condition 1 – the output symbol $\beta(t)$ must be produced without delay after receipt of the input symbol $\alpha(t)$;
- $\beta(2t) = \beta(2t+1) = \alpha(t)$ (β "doubles" the position of each symbol of α) violates condition 2 – we need to record an unboundedly increasing number of symbols for future use;

FORMALIZATION OF THE PROBLEM (CONT'D)

EXAMPLES

- $\beta(t) = \alpha(2t)$ (β returns the elements at even positions of α) violates condition 1 – the output symbol $\beta(t)$ must be produced without delay after receipt of the input symbol $\alpha(t)$;
- $\beta(2t) = \beta(2t + 1) = \alpha(t)$ (β "doubles" the position of each symbol of α) violates condition 2 – we need to record an unboundedly increasing number of symbols for future use;
- $\beta = 111\dots$, if α features an infinite number of occurrences of 1; otherwise, $\beta = 000\dots$ violates condition 1 as well – the first symbol of the output sequence β cannot be determined on the basis of any finite prefix of α.

Formalization of the problem (cont'd)

A finite state machine

- A Mealy automaton (input-output automaton or transducer) $\mathcal{M}$: a finite state automaton with an output function $\tau : S \times \Sigma \to \Gamma$, where S is a finite set of states, Σ is a finite input alphabet, and Γ is a finite output alphabet.

FORMALIZATION OF THE PROBLEM (CONT'D)

A FINITE STATE MACHINE

- A Mealy automaton (input-output automaton or transducer) $\mathcal{M}$: a finite state automaton with an output function $\tau : S \times \Sigma \to \Gamma$, where S is a finite set of states, Σ is a finite input alphabet, and Γ is a finite output alphabet.
- Given an input sequence

$$\alpha = \alpha(1)\alpha(2)\cdots,$$

the output sequence computed by $\mathcal{M}$ is

$$\mathcal{M}(\alpha) = \beta = \beta(1)\beta(2)\cdots,$$

where $\beta(t) = \tau(\delta^*(s_0, \alpha(0)\cdots\alpha(t-1)), \alpha(t))$
(δ^* extends the transition function as follows:
$\delta^*(s, \epsilon) = s$; $\delta^*(s, wa) = \delta(\delta^*(s, w), a)$, for $w \in \Sigma^*$ and $a \in \Sigma$).

FORMALIZATION OF THE PROBLEM (CONT'D)

A FINITE STATE MACHINE

- A Mealy automaton (input-output automaton or transducer) $\mathcal{M}$: a finite state automaton with an output function $\tau : S \times \Sigma \to \Gamma$, where S is a finite set of states, Σ is a finite input alphabet, and Γ is a finite output alphabet.
- Given an input sequence

$$\alpha = \alpha(1)\alpha(2)\cdots ,$$

the output sequence computed by $\mathcal{M}$ is

$$\mathcal{M}(\alpha) = \beta = \beta(1)\beta(2)\cdots ,$$

where $\beta(t) = \tau(\delta^*(s_0, \alpha(0)\cdots\alpha(t-1)), \alpha(t))$
(δ^* extends the transition function as follows:
$\delta^*(s, \epsilon) = s$; $\delta^*(s, wa) = \delta(\delta^*(s, w), a)$, for $w \in \Sigma^*$ and $a \in \Sigma$).
- It satisfies the conditions on transformations.

FORMALIZATION OF THE PROBLEM (CONT'D)

THE SPECIFICATION LANGUAGE

- $S1S$: the monadic second-order logic of one successor denoted by $(\omega, +1)$.

FORMALIZATION OF THE PROBLEM (CONT'D)

THE SPECIFICATION LANGUAGE

- *S1S*: the monadic second-order logic of one successor denoted by $(\omega, +1)$.
- The ordering relation $<$ is (second-order) definable in terms of the successor function $+1$: $s < t$ if and only if t belongs to each set that includes $s + 1$ and is closed under successor.

FORMALIZATION OF THE PROBLEM (CONT'D)

THE SPECIFICATION LANGUAGE

- $S1S$: the monadic second-order logic of one successor denoted by $(\omega, +1)$.
- The ordering relation $<$ is (second-order) definable in terms of the successor function $+1$: $s < t$ if and only if t belongs to each set that includes $s + 1$ and is closed under successor.
- For the sake of simplicity, we will only consider Boolean input and output alphabets, that is, $\{0, 1\}$.

FORMALIZATION OF THE PROBLEM (CONT'D)

THE SPECIFICATION LANGUAGE

- $S1S$: the monadic second-order logic of one successor denoted by $(\omega, +1)$.
- The ordering relation $<$ is (second-order) definable in terms of the successor function $+1$: $s < t$ if and only if t belongs to each set that includes $s + 1$ and is closed under successor.
- For the sake of simplicity, we will only consider Boolean input and output alphabets, that is, $\{0, 1\}$.
- The $S1S$-formulas $\varphi(X, Y)$ we will take into consideration talk about sequences $\alpha \in \{0, 1\}^\omega$ and $\beta \in \{0, 1\}^\omega$.

  The free variable X identifies those positions where α takes value 1, while the free variable Y identifies those where β takes value 1.

  We denote the interpretations of X and Y induced by α and β by P_α and P_β, respectively.

FORMALIZATION OF THE PROBLEM (CONT'D)

CHURCH'S PROBLEM

Church's problem can be precisely stated as follows:

FORMALIZATION OF THE PROBLEM (CONT'D)

CHURCH'S PROBLEM

Church's problem can be precisely stated as follows:

given an S1S-formula $\varphi(X, Y)$*, build a Mealy automaton* $\mathcal{M}$*, with input alphabet* $\Sigma = \{0, 1\}$ *and output alphabet* $\Gamma = \{0, 1\}$*, such that, for every input sequence* $\alpha \in \{0, 1\}^\omega$*,* $\mathcal{M}$ *generates an output sequence* $\beta \in \{0, 1\}^\omega$ *such that* $(\omega, +1) \models \varphi[P_\alpha, P_\beta]$ *(or it answers that such an automaton does not exist).*

It can be easily generalized to an input alphabet $\Sigma = \{0, 1\}^{m_1}$ and/or to an output alphabet $\Gamma = \{0, 1\}^{m_2}$.

FORMALIZATION OF THE PROBLEM (CONT'D)

CHURCH'S PROBLEM

Church's problem can be precisely stated as follows:

given an S1S-formula $\varphi(X, Y)$*, build a Mealy automaton* $\mathcal{M}$*, with input alphabet* $\Sigma = \{0, 1\}$ *and output alphabet* $\Gamma = \{0, 1\}$*, such that, for every input sequence* $\alpha \in \{0, 1\}^\omega$*,* $\mathcal{M}$ *generates an output sequence* $\beta \in \{0, 1\}^\omega$ *such that* $(\omega, +1) \models \varphi[P_\alpha, P_\beta]$ *(or it answers that such an automaton does not exist).*

It can be easily generalized to an input alphabet $\Sigma = \{0, 1\}^{m_1}$ and/or to an output alphabet $\Gamma = \{0, 1\}^{m_2}$.

A finite state winning strategy for an infinite game: according to a game-theoretic interpretation, a Mealy automaton can be viewed as the definition of a winning strategy for player B/β (Bob) that replies to the moves of player A/α (Alice).

THE SOLUTION BY BÜCHI-LANDWEBER

- The solution by Büchi-Landweber is based on a series of transformations that, starting from the logical characterization of the problem, allow one to first replace it with a characterization based on automata on infinite words (Muller automata) and then with a characterization based on infinite games (which are played on the transition graph of the automaton).

THE SOLUTION BY BÜCHI-LANDWEBER

- The solution by Büchi-Landweber is based on a series of transformations that, starting from the logical characterization of the problem, allow one to first replace it with a characterization based on automata on infinite words (Muller automata) and then with a characterization based on infinite games (which are played on the transition graph of the automaton).

S1S

$\Downarrow$

(Deterministic) Muller automata

$\Downarrow$

Muller games

$\Downarrow$

Parity games

FROM LOGIC TO (MULLER) AUTOMATA

- We first transform an S1S specification $\varphi(X, Y)$ into a deterministic Muller automaton $\mathcal{A}$, that recognizes infinite words γ in $(\{0,1\} \times \{0,1\})^\omega$, in such a way that
  - $\mathcal{A}$ accepts γ if and only if $(\omega, +1) \vDash \varphi[\overline{P_\gamma}]$.

FROM LOGIC TO (MULLER) AUTOMATA

- We first transform an S1S specification $\varphi(X, Y)$ into a deterministic Muller automaton $\mathcal{A}$, that recognizes infinite words γ in $(\{0,1\} \times \{0,1\})^\omega$, in such a way that
  - $\mathcal{A}$ accepts γ if and only if $(\omega, +1) \models \varphi[\overline{P_\gamma}]$.
- From automata theory, we know that:
  (i) S1S formulas are equivalent to nondeterministic Büchi automata (NBA) and NBA are equivalent to deterministic Muller automata (DMA);
  (ii) these transformations are effective.
  - Muller acceptance condition: given a collection of sets of states $\mathcal{F} = \{F_1, ..., F_k\}$, a computation σ by $\mathcal{A}$ is successful if the set of states that occur infinitely often in σ belongs to $\mathcal{F}$.

FROM LOGIC TO (MULLER) AUTOMATA

- We first transform an S1S specification $\varphi(X, Y)$ into a deterministic Muller automaton $\mathcal{A}$, that recognizes infinite words γ in $(\{0,1\} \times \{0,1\})^\omega$, in such a way that
  - $\mathcal{A}$ accepts γ if and only if $(\omega, +1) \models \varphi[\overline{P_\gamma}]$.
- From automata theory, we know that:
  (i) S1S formulas are equivalent to nondeterministic Büchi automata (NBA) and NBA are equivalent to deterministic Muller automata (DMA);
  (ii) these transformations are effective.
  - Muller acceptance condition: given a collection of sets of states $\mathcal{F} = \{F_1, ..., F_k\}$, a computation σ by $\mathcal{A}$ is successful if the set of states that occur infinitely often in σ belongs to $\mathcal{F}$.
- Remark: the above transformations are computable but extremely expensive (in terms of resources), as $|\mathcal{A}_\varphi|$ cannot be bounded by a function elementary in the size of $|\varphi|$.

FROM (MULLER) AUTOMATA TO (MULLER) GAMES

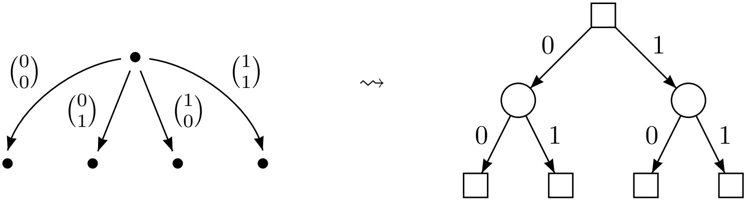
- The automaton at the previous step is then transformed into the graph of a two player game.

FROM (MULLER) AUTOMATA TO (MULLER) GAMES

- The automaton at the previous step is then transformed into the graph of a two player game.
- Such a transformation allows one to make the contributions (in terms of bits) of the two players explicit.

FROM (MULLER) AUTOMATA TO (MULLER) GAMES

- The automaton at the previous step is then transformed into the graph of a two player game.
- Such a transformation allows one to make the contributions (in terms of bits) of the two players explicit.



- □ = states of A (states of the Muller automaton)
- ◯ = states of B

NOTATIONAL CONVENTIONS

- For each A-state q and each bit b, we introduce a transition labeled by b to a new B-state that we call (q, b).

NOTATIONAL CONVENTIONS

- For each A-state q and each bit b, we introduce a transition labeled by b to a new B-state that we call (q, b).
- For each bit c, we introduce a transition labeled by c from the B-state (q, b) to the A-state p whenever in the original Muller automaton it is possible to move from q to p via a transition labeled by the pair of bits (b, c).

Notational conventions

- For each A-state q and each bit b, we introduce a transition labeled by b to a new B-state that we call (q, b).
- For each bit c, we introduce a transition labeled by c from the B-state (q, b) to the A-state p whenever in the original Muller automaton it is possible to move from q to p via a transition labeled by the pair of bits (b, c).
- For such a state p, we define c as the output bit and we denote it by $out(q, b, p)$ (if both transitions exiting from (q, b) lead to the same state p, we put by convention $out(q, b, p) = 0$).

SOME REMARKS

Some effects of the replacement of automata by games.

SOME REMARKS

Some effects of the replacement of automata by games.

- The game-theoretic perspective introduces a symmetry between input and output: player A, who provides the input, aims at falsifying the specification; player B, who provides the output, aims at satisfying it. As we will see, it is possible to prove that one of the two players has a winning strategy (a feature which was hidden in the original formulation of the problem).

SOME REMARKS

Some effects of the replacement of automata by games.

- The game-theoretic perspective introduces a symmetry between input and output: player A, who provides the input, aims at falsifying the specification; player B, who provides the output, aims at satisfying it. As we will see, it is possible to prove that one of the two players has a winning strategy (a feature which was hidden in the original formulation of the problem).
- The game-theoretic perspective does not assign a special role to the initial state: the problem is to determine for each state which player has a winning strategy in a game that starts from such a state.

SOME REMARKS

Some effects of the replacement of automata by games.

- The game-theoretic perspective introduces a symmetry between input and output: player A, who provides the input, aims at falsifying the specification; player B, who provides the output, aims at satisfying it. As we will see, it is possible to prove that one of the two players has a winning strategy (a feature which was hidden in the original formulation of the problem).
- The game-theoretic perspective does not assign a special role to the initial state: the problem is to determine for each state which player has a winning strategy in a game that starts from such a state.
- The labels associated with the transitions can be initially ignored, as the winning conditions are given in terms of visited states, and only subsequently reintroduced, when the Mealy automaton must be synthesized.

GAME GRAPH AND MEALY AUTOMATON

An important remark.

Do not confuse the states of the game graph with the states of the (finite state) Mealy automaton: the Mealy automaton works on the game graph, but its states are not the states of the game graph.

As we will see, to solve Church's problem we need to combine in suitable way the states of the Mealy automaton and those of the game graph.

THE SOLUTION

In the following, we show how to obtain a solution to Church's problem in two steps, starting from a finite game graph with Muller winning conditions:

1. to establish whether or not B wins;
2. in case of a positive answer, to provide a (finite state) winning strategy.

2. INFINITE GAMES AND BÜCHI-LANDWEBER THEOREM

Infinite games
Büchi-Landweber Theorem

Infinite games

- The game graph (arena) is a graph $G = (Q, Q_A, E)$, with $Q_A \subseteq Q$ and $E \subseteq Q \times Q$, where $\forall q \in Q : qE \neq \varnothing$ (no deadlock). Let $Q_B = Q \setminus Q_A$. We will only consider finite game graphs. Moreover, by construction, each edge leads from a state in Q_A to a state in Q_B or vice versa. Nevertheless, the results we are going to provide do not depend on such an assumption.

INFINITE GAMES

- The game graph (arena) is a graph $G = (Q, Q_A, E)$, with $Q_A \subseteq Q$ and $E \subseteq Q \times Q$, where $\forall q \in Q : qE \neq \varnothing$ (no deadlock). Let $Q_B = Q \setminus Q_A$. We will only consider finite game graphs. Moreover, by construction, each edge leads from a state in Q_A to a state in Q_B or vice versa. Nevertheless, the results we are going to provide do not depend on such an assumption.
- A play on G from q is an infinite path ρ on G with initial state q (infinite games). We assume A to choose the next state when we are in a Q_A state and b to choose it when we are in a Q_B state.

INFINITE GAMES

- The game graph (arena) is a graph $G = (Q, Q_A, E)$, with $Q_A \subseteq Q$ and $E \subseteq Q \times Q$, where $\forall q \in Q : qE \neq \varnothing$ (no deadlock). Let $Q_B = Q \setminus Q_A$. We will only consider finite game graphs. Moreover, by construction, each edge leads from a state in Q_A to a state in Q_B or vice versa. Nevertheless, the results we are going to provide do not depend on such an assumption.
- A play on G from q is an infinite path ρ on G with initial state q (infinite games). We assume A to choose the next state when we are in a Q_A state and b to choose it when we are in a Q_B state.
- A game is a pair (G, W), where $G = (Q, Q_A, E)$ is a game graph and $W \subseteq Q^\omega$ is the winning condition for player B. Player B wins the play $\rho = q_0 q_1 q_2 \cdots$ if $\rho \in W$, otherwise A wins ρ.

INFINITE GAMES

- The game graph (arena) is a graph $G = (Q, Q_A, E)$, with $Q_A \subseteq Q$ and $E \subseteq Q \times Q$, where $\forall q \in Q : qE \neq \varnothing$ (no deadlock). Let $Q_B = Q \setminus Q_A$. We will only consider finite game graphs. Moreover, by construction, each edge leads from a state in Q_A to a state in Q_B or vice versa. Nevertheless, the results we are going to provide do not depend on such an assumption.
- A play on G from q is an infinite path ρ on G with initial state q (infinite games). We assume A to choose the next state when we are in a Q_A state and b to choose it when we are in a Q_B state.
- A game is a pair (G, W), where $G = (Q, Q_A, E)$ is a game graph and $W \subseteq Q^\omega$ is the winning condition for player B. Player B wins the play $\rho = q_0 q_1 q_2 \cdots$ if $\rho \in W$, otherwise A wins ρ.
- We are interested in winning conditions which can be expressed in a finite way (finitely describable).

MULLER GAMES, WEAK MULLER GAMES, AND REACHABILITY GAMES

- Muller games: the winning condition is a collection of sets of states $\mathcal{F} \subseteq 2^{Q}$ such that B wins ρ if and only if $\text{Inf}(\rho) \in \mathcal{F}$.

MULLER GAMES, WEAK MULLER GAMES, AND REACHABILITY GAMES

- Muller games: the winning condition is a collection of sets of states $\mathcal{F} \subseteq 2^Q$ such that B wins ρ if and only if $\text{Inf}(\rho) \in \mathcal{F}$.
- Weak Muller games: there exists a weak version of the winning condition of Muller games (Staiger-Wagner condition), according to which B wins ρ if and only if $Occ(\rho) \in \mathcal{F}$, where $Occ(\rho) = \{q \in Q : \exists i(\rho(i) = q\})$.

MULLER GAMES, WEAK MULLER GAMES, AND REACHABILITY GAMES

- Muller games: the winning condition is a collection of sets of states $\mathcal{F} \subseteq 2^Q$ such that B wins ρ if and only if $\mathrm{Inf}(\rho) \in \mathcal{F}$.
- Weak Muller games: there exists a weak version of the winning condition of Muller games (Staiger-Wagner condition), according to which B wins ρ if and only if $Occ(\rho) \in \mathcal{F}$, where $Occ(\rho) = \{q \in Q : \exists i(\rho(i) = q\})$.
- Reachability games: given $F \subseteq Q$ in (Q, Q_A, E), B wins ρ if some state in ρ belongs to F.

MULLER GAMES, WEAK MULLER GAMES, AND REACHABILITY GAMES

- Muller games: the winning condition is a collection of sets of states $\mathcal{F} \subseteq 2^Q$ such that B wins ρ if and only if $\mathrm{Inf}(\rho) \in \mathcal{F}$.
- Weak Muller games: there exists a weak version of the winning condition of Muller games (Staiger-Wagner condition), according to which B wins ρ if and only if $Occ(\rho) \in \mathcal{F}$, where $Occ(\rho) = \{q \in Q : \exists i(\rho(i) = q\})$.
- Reachability games: given $F \subseteq Q$ in (Q, Q_A, E), B wins ρ if some state in ρ belongs to F.

  Reachability games can be easily expressed in terms of Staiger-Wagner condition: $\mathcal{F} = \{R \subseteq Q : R \cap F \neq \emptyset\}$.

STRATEGIES, WINNING STRATEGIES, WINNING REGIONS, AND DETERMINED GAMES

- A strategy for a player is a mapping $f : Q^+ \rightarrow Q$ such that, given the history of the game up a certain state (under his/her control), specifies his/her behavior at the next step.

STRATEGIES, WINNING STRATEGIES, WINNING REGIONS, AND DETERMINED GAMES

- A strategy for a player is a mapping $f : Q^+ \rightarrow Q$ such that, given the history of the game up a certain state (under his/her control), specifies his/her behavior at the next step.
- A strategy f for player B from q is a winning strategy if each play from q, played according to f, is won by player B.

STRATEGIES, WINNING STRATEGIES, WINNING REGIONS, AND DETERMINED GAMES

- A strategy for a player is a mapping $f : Q^+ \rightarrow Q$ such that, given the history of the game up a certain state (under his/her control), specifies his/her behavior at the next step.
- A strategy f for player B from q is a winning strategy if each play from q, played according to f, is won by player B.
- $W_B := \{q \in Q \mid B \text{ wins starting from } q\}$ is said the winning region of B (the same for A). Obviously, $W_A \cap W_B = \emptyset$.

STRATEGIES, WINNING STRATEGIES, WINNING REGIONS, AND DETERMINED GAMES

- A strategy for a player is a mapping $f : Q^+ \rightarrow Q$ such that, given the history of the game up a certain state (under his/her control), specifies his/her behavior at the next step.
- A strategy f for player B from q is a winning strategy if each play from q, played according to f, is won by player B.
- $W_B := \{q \in Q \mid B \text{ wins starting from } q\}$ is said the winning region of B (the same for A). Obviously, $W_A \cap W_B = \emptyset$.
- If $W_A \cup W_B = Q$, we say that the game is determined.

SOLUTION OF A GAME AND POSITIONAL STRATEGIES

- The solution of a game (G, W), with $G = (Q, Q_A, E)$ and W finitely describable, consists of two steps:

  (i) to establish, for each $q \in Q$, if $q \in W_B$ or $q \in W_A$;
  (ii) to build a (finitely describable) winning strategy starting from q (for B, if $q \in W_B$; for A, otherwise).

SOLUTION OF A GAME AND POSITIONAL STRATEGIES

- The solution of a game (G, W), with $G = (Q, Q_A, E)$ and W finitely describable, consists of two steps:

  (i) to establish, for each $q \in Q$, if $q \in W_B$ or $q \in W_A$;
  (ii) to build a (finitely describable) winning strategy starting from q (for B, if $q \in W_B$; for A, otherwise).

We distinguish two types of strategy: positional and finite state.

SOLUTION OF A GAME AND POSITIONAL STRATEGIES

- The solution of a game (G, W), with $G = (Q, Q_A, E)$ and W finitely describable, consists of two steps:

  (i) to establish, for each $q \in Q$, if $q \in W_B$ or $q \in W_A$;
  (ii) to build a (finitely describable) winning strategy starting from q (for B, if $q \in W_B$; for A, otherwise).

  We distinguish two types of strategy: positional and finite state.

- A strategy $f : Q^+ \rightarrow Q$ is positional if the value of $f(q_1 \cdots q_k)$ only depends on the current state q_k. A positional strategy for B is a mapping $f : Q_B \rightarrow Q$ (the same for A).

  In graph-theoretic terms, a positional strategy for B can be expressed as a subset of edges of G, which includes all edges exiting from states in Q_A and one edge exiting from states in Q_B (the one identified by the function).

FINITE STATE STRATEGIES AND COMPUTED STRATEGIES

- A strategy $f : Q^+ \to Q$ on a finite set of states Q can be viewed as a function on words.

FINITE STATE STRATEGIES AND COMPUTED STRATEGIES

- A strategy $f : Q^+ \to Q$ on a finite set of states Q can be viewed as a function on words.
- Formally, f is a finite state strategy if it can be computed by a Mealy automaton of the form $\mathcal{S} = (S, Q, Q, s_0, \delta, \tau)$, where S is a finite set of states, Q is both the input and output alphabet, $s_0 \in S$ is the initial state, $\delta : S \times Q \to S$, and $\tau : S \times Q_A \to Q$, for A, and $\tau : S \times Q_B \to Q$, for B.

FINITE STATE STRATEGIES AND COMPUTED STRATEGIES

- A strategy $f : Q^+ \to Q$ on a finite set of states Q can be viewed as a function on words.
- Formally, f is a finite state strategy if it can be computed by a Mealy automaton of the form $\mathcal{S} = (S, Q, Q, s_0, \delta, \tau)$, where S is a finite set of states, Q is both the input and output alphabet, $s_0 \in S$ is the initial state, $\delta : S \times Q \to S$, and $\tau : S \times Q_A \to Q$, for A, and $\tau : S \times Q_B \to Q$, for B.
- The strategy $f_{\mathcal{S}}$ computed by $\mathcal{S}$ can be defined by $f_{\mathcal{S}}(q_0 \cdots q_k) = \tau(\delta^*(s_0, q_0 \cdots q_{k-1}), q_k)$, where $\delta^*(s, w)$ is the state reached by $\mathcal{S}$ starting from s on the input word w and τ is chosen by the player who is responsible for q_k.

Büchi-Landweber Theorem

Theorem (Weak Muller games)

Weak Muller games are determined and for each weak Muller game $(G, \mathcal{F})$*, where G has n states, the winning regions for the two players can be effectively determined and it is possible to build, for each state q in G, a finite state winning strategy from q (for the winning player) making use of a memory with* 2^n *states.*

Büchi-Landweber Theorem

Theorem (Weak Muller games)

Weak Muller games are determined and for each weak Muller game $(G, \mathcal{F})$*, where G has n states, the winning regions for the two players can be effectively determined and it is possible to build, for each state q in G, a finite state winning strategy from q (for the winning player) making use of a memory with* 2^n *states.*

Theorem (Muller games / Büchi-Landweber Theorem)

Muller games are determined and for each Muller game $(G, \mathcal{F})$*, where G has n states, the winning regions for the two players can be effectively determined and it is possible to build, for each state q in G, a finite state winning strategy from q (for the winning player) making use of a memory with* $n! \cdot n$ *states.*

A SOLUTION TO CHURCH'S PROBLEM

The fundamental steps:

A SOLUTION TO CHURCH'S PROBLEM

The fundamental steps:

1. given an S1S-formula $\varphi(X, Y)$, we transform it into a Muller automaton $\mathcal{M}$;

A SOLUTION TO CHURCH'S PROBLEM

The fundamental steps:

1. given an S1S-formula $\varphi(X, Y)$, we transform it into a Muller automaton $\mathcal{M}$;
2. then, we transform $\mathcal{M}$ into a game graph G with Muller winning condition (G inherits its initial state from $\mathcal{M}$);

A SOLUTION TO CHURCH'S PROBLEM

The fundamental steps:

1. given an S1S-formula $\varphi(X, Y)$, we transform it into a Muller automaton $\mathcal{M}$;
2. then, we transform $\mathcal{M}$ into a game graph G with Muller winning condition (G inherits its initial state from $\mathcal{M}$);
3. Büchi-Landweber Theorem makes it possible to determine the winning regions and to establish whether the initial state of the game belongs to W_B; in such a case, we build the Mealy automaton $\mathcal{S}$ which realizes the winning strategy, starting from the initial state ($\mathcal{S}$ is called the strategy automaton);

A SOLUTION TO CHURCH'S PROBLEM

The fundamental steps:

1. given an S1S-formula $\varphi(X, Y)$, we transform it into a Muller automaton $\mathcal{M}$;
2. then, we transform $\mathcal{M}$ into a game graph G with Muller winning condition (G inherits its initial state from $\mathcal{M}$);
3. Büchi-Landweber Theorem makes it possible to determine the winning regions and to establish whether the initial state of the game belongs to W_B; in such a case, we build the Mealy automaton $\mathcal{S}$ which realizes the winning strategy, starting from the initial state ($\mathcal{S}$ is called the strategy automaton);
4. the Mealy automaton $\mathcal{A}$, that solves Church's problem, is obtained from the product of the automata $\mathcal{M}$ and $\mathcal{S}$.

It is worth pointing out that Büchi-Landweber Theorem is exploited only at step 3.

THE LAST STEP IN DETAIL

1. The state space of $\mathcal{A}$ is $Q \times S$, where Q is the set of states of the Muller automaton $\mathcal{M}$ and S is the set of states of the strategy automaton $\mathcal{S}$, and its initial state is (q_0, s_0);

The last step in detail

1. The state space of $\mathcal{A}$ is $Q \times S$, where Q is the set of states of the Muller automaton $\mathcal{M}$ and S is the set of states of the strategy automaton $\mathcal{S}$, and its initial state is (q_0, s_0);
2. a transition (of the Mealy automaton) must be specified for each state (q, s) and each input bit b, that is, an output bit b' and a new state (q', s');

THE LAST STEP IN DETAIL

1. The state space of $\mathcal{A}$ is $Q \times S$, where Q is the set of states of the Muller automaton $\mathcal{M}$ and S is the set of states of the strategy automaton $\mathcal{S}$, and its initial state is (q_0, s_0);
2. a transition (of the Mealy automaton) must be specified for each state (q, s) and each input bit b, that is, an output bit b' and a new state (q', s');
3. to this end, the state $q^* = (q, b)$ of the game graph G and the state $s^* = \delta(s, q^*)$ of the strategy automaton $\mathcal{S}$ associated with it must be computed;

THE LAST STEP IN DETAIL

1. The state space of $\mathcal{A}$ is $Q \times S$, where Q is the set of states of the Muller automaton $\mathcal{M}$ and S is the set of states of the strategy automaton $\mathcal{S}$, and its initial state is (q_0, s_0);
2. a transition (of the Mealy automaton) must be specified for each state (q, s) and each input bit b, that is, an output bit b' and a new state (q', s');
3. to this end, the state $q^* = (q, b)$ of the game graph G and the state $s^* = \delta(s, q^*)$ of the strategy automaton $\mathcal{S}$ associated with it must be computed;
4. the output function of $\mathcal{S}$ returns the state $q' = \tau(s^*, q^*)$ of the game graph G, while its transition function returns the new memory state $s' = \delta(s^*, q')$;

THE LAST STEP IN DETAIL

1. The state space of $\mathcal{A}$ is $Q \times S$, where Q is the set of states of the Muller automaton $\mathcal{M}$ and S is the set of states of the strategy automaton $\mathcal{S}$, and its initial state is (q_0, s_0);
2. a transition (of the Mealy automaton) must be specified for each state (q, s) and each input bit b, that is, an output bit b' and a new state (q', s');
3. to this end, the state $q^* = (q, b)$ of the game graph G and the state $s^* = \delta(s, q^*)$ of the strategy automaton $\mathcal{S}$ associated with it must be computed;
4. the output function of $\mathcal{S}$ returns the state $q' = \tau(s^*, q^*)$ of the game graph G, while its transition function returns the new memory state $s' = \delta(s^*, q')$;
5. the output bit b' is the value $out(q, b, q')$ associated with the transition from $q^* = (q, b)$ to q'.

Remark: the memory of $\mathcal{A}$ combines the state space of the Muller automaton $\mathcal{M}$ and the state space of the strategy automaton $\mathcal{S}$ (see item 1).

Reachability games

Theorem

A reachability game (G, F), with $G = (Q, Q_A, E)$ and $F \subseteq Q$, is determined and both the winning regions W_A and W_B for players A and B, respectively, and the corresponding positional winning strategies are computable.

REACHABILITY GAMES

Theorem

A reachability game (G, F), with $G = (Q, Q_A, E)$ and $F \subseteq Q$, is determined and both the winning regions W_A and W_B for players A and B, respectively, and the corresponding positional winning strategies are computable.

Proof.

For $i = 0, 1, \ldots$, compute the vertices starting from which player B can force a visit in F in at most i moves (i-the attractor $Attr_B^i(F)$).

The sequence $Attr_B^0(F)(= F) \subseteq Attr_B^1(F) \subseteq Attr_B^2(F) \ldots$ becomes stationary for some index $k \leq |Q|$. We define $Attr_B(F) = \bigcup_{i=0}^{|Q|} Attr_B^i(F)$.

It can be easily proved that $W_B = Attr_B(F)$. □

Weak Muller games

It is possible to show that the winning condition for weak Muller games (player B wins a play ρ if and only if $Occ(\rho) \in \mathcal{F}$, that is, the collection of the states visited by ρ is one of the set in $\mathcal{F}$) can be expressed as Boolean combinations of reachability conditions.

WEAK MULLER GAMES

It is possible to show that the winning condition for weak Muller games (player B wins a play ρ if and only if $Occ(\rho) \in \mathcal{F}$, that is, the collection of the states visited by ρ is one of the set in $\mathcal{F}$) can be expressed as Boolean combinations of reachability conditions.

In general, positional strategies do not suffice to win weak Muller games. In some cases, indeed, it is necessary to remember the states that have been already visited.

WEAK MULLER GAMES

It is possible to show that the winning condition for weak Muller games (player B wins a play ρ if and only if $Occ(\rho) \in \mathcal{F}$, that is, the collection of the states visited by ρ is one of the set in $\mathcal{F}$) can be expressed as Boolean combinations of reachability conditions.

In general, positional strategies do not suffice to win weak Muller games. In some cases, indeed, it is necessary to remember the states that have been already visited.

Solution: a Mealy automaton $\mathcal{S}$ with the set Q of the states of the game as its input alphabet, the powerset of Q as the set of its states ($2^{|Q|}$ states), and $\emptyset$ as the initial state.
The idea of the appearance record: on the input word $q_1, \ldots, q_k$, $\mathcal{S}$ reaches the state $\{q_1, \ldots, q_k\}$ ($\delta(R, p) = R \cup \{p\}$).

The rewriting of weak Muller games as weak parity games

It is possible to associate a number (color) $c(R)$ with each $R \subseteq Q$ that codifies two pieces of information: the size of R and the membership (or not) of R to $\mathcal{F}$.

Formally, $c(R) = 2 \cdot |R|$ if $R \in \mathcal{F}$ and $c(R) = 2 \cdot |R| - 1$ if $R \notin \mathcal{F}$.

THE REWRITING OF WEAK MULLER GAMES AS WEAK PARITY GAMES

It is possible to associate a number (color) $c(R)$ with each $R \subseteq Q$ that codifies two pieces of information: the size of R and the membership (or not) of R to $\mathcal{F}$.

Formally, $c(R) = 2 \cdot |R|$ if $R \in \mathcal{F}$ and $c(R) = 2 \cdot |R| - 1$ if $R \notin \mathcal{F}$.

Let ρ be a play and $R_0, R_1, R_2, \ldots$ be the associated sequence of appearance records.

It holds that $Occ(\rho) \in \mathcal{F}$ if and only if the maximum color of the sequence $c(R_0), c(R_1), c(R_2), \ldots$ is even.

The rewriting of weak Muller games as weak parity games

It is possible to associate a number (color) $c(R)$ with each $R \subseteq Q$ that codifies two pieces of information: the size of R and the membership (or not) of R to $\mathcal{F}$.

Formally, $c(R) = 2 \cdot |R|$ if $R \in \mathcal{F}$ and $c(R) = 2 \cdot |R| - 1$ if $R \notin \mathcal{F}$.

Let ρ be a play and $R_0, R_1, R_2, \ldots$ be the associated sequence of appearance records.

It holds that $Occ(\rho) \in \mathcal{F}$ if and only if the maximum color of the sequence $c(R_0), c(R_1), c(R_2), \ldots$ is even.

A weak Muller game can be transformed into a weak parity game (game simulation).

GAME SIMULATION

PROOF OF BÜCHI-LANDWEBER THEOREM

We say that a game (G, W), with $G = (Q, Q_A, E)$, is simulated by a game (G', W'), with $G' = (Q', Q'_A, E')$, if there exists a finite state automaton $\mathcal{S} = (S, Q, s_0, \delta)$, devoid of final states, such that:

- $Q' = S \times Q$;
- $Q'_A = S \times Q_A$;
- $((r, p), (s, q)) \in E'$ if and only if $(p, q) \in E$ and $\delta(r, p) = s$, from which it follows that a play $\rho = q_0 q_1 \ldots$ in G induces a play $\rho' = (s_0, q_0)(\delta(s_0, q_0), q_1) \ldots$ in G';
- a play ρ on G belongs to W if and only if the corresponding play ρ' on G' belongs to W'.

Whenever the above conditions hold, we write $(G, W) \leq_{\mathcal{S}} (G', W')$.

GAME SIMULATION (CONT'D)

PROOF OF BÜCHI-LANDWEBER THEOREM

Consequence: positional strategies for G' can be easily transformed into finite state strategies for G (a Mealy automaton). The latter strategies can be realized by automata $\mathcal{S}$ enriched with an output function obtained from the positional strategy for G'.

Lemma

If there exists a positional winning strategy for player B in (G', W') from (s_0, q), then player B has a finite state winning strategy from q in (G, W).

Proof.

We extend the automaton $\mathcal{S}$ with an output function extracted from the winning strategy $\sigma : Q'_B \to Q'$. To this end, it suffices to define $\tau : S \times Q_B \to Q$ as $\tau(s, q) := \pi_2(\sigma(s, q))$, where $\pi_2(\sigma(s, q))$ is simply the projection on the second component of $\sigma(s, q)$. □

From Muller to parity games

Proof of Büchi-Landweber Theorem

- Muller games can be simulated by parity games by means of the LAR (Latest Appearance Record) structure.

From Muller to parity games

Proof of Büchi-Landweber Theorem

- Muller games can be simulated by parity games by means of the LAR (Latest Appearance Record) structure.
- Intuitively, a LAR represents the sequence of states encountered during a play, ordered according to their last occurrence / appearance. If the current state was already visited in the past, then it is moved from the position h (called hit) it occupies in the current LAR to the first position of the new LAR.

FROM MULLER TO PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- Muller games can be simulated by parity games by means of the LAR (Latest Appearance Record) structure.
- Intuitively, a LAR represents the sequence of states encountered during a play, ordered according to their last occurrence / appearance. If the current state was already visited in the past, then it is moved from the position h (called hit) it occupies in the current LAR to the first position of the new LAR.
- Given a LAR $((i_1 \dots i_r), h)$, its hitting set is the set $\{i_1, \dots, i_h\}$ of the states which were encountered up to the hit h (including position h).

AN EXAMPLE OF THE USE OF LAR

PROOF OF BÜCHI-LANDWEBER THEOREM

State	LAR	Hitting set
A	(A,0)	{}
C	(CA,0)	{}
C	(CA,1)	$\{C\}$
D	(DCA,0)	{}
B	(BDCA,0)	{}
D	(DBCA,2)	$\{B, D\}$
C	(CDBA,3)	$\{B, C, D\}$
D	(DCBA,2)	$\{C, D\}$
D	(DCBA,1)	$\{D\}$

Let us consider the 7-th row of the table. The hitting set $\{B, C, D\}$ consists of all and only those states which have been encountered in between the last two occurrences of C (C included).

PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- Let ρ be a sequence over Q and ρ' be the corresponding sequence of LARs. The set $\text{Inf}(\rho)$ coincides with the hitting set H of the maximum hit h that occurs infinitely often in ρ'.

Parity games

Proof of Büchi-Landweber Theorem

- Let ρ be a sequence over Q and ρ' be the corresponding sequence of LARs. The set $\mathrm{Inf}(\rho)$ coincides with the hitting set H of the maximum hit h that occurs infinitely often in ρ'.
- The winning condition for the play ρ of a Muller game can be reformulated as follows: *H belongs to $\mathcal{F}$*.

PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- Let ρ be a sequence over Q and ρ' be the corresponding sequence of LARs. The set $\mathrm{Inf}(\rho)$ coincides with the hitting set H of the maximum hit h that occurs infinitely often in ρ'.
- The winning condition for the play ρ of a Muller game can be reformulated as follows: *H belongs to $\mathcal{F}$*.
- The winning condition for Muller games can be redefined in terms of a suitable coloring of LAR.

PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- Let ρ be a sequence over Q and ρ' be the corresponding sequence of LARs. The set $\mathrm{Inf}(\rho)$ coincides with the hitting set H of the maximum hit h that occurs infinitely often in ρ'.
- The winning condition for the play ρ of a Muller game can be reformulated as follows: *H belongs to $\mathcal{F}$*.
- The winning condition for Muller games can be redefined in terms of a suitable coloring of LAR.
- Parity condition: B wins ρ' if and only if the greatest color that occurs infinitely often in $c(\rho'(0))c(\rho'(1))\ldots$ is even.

PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- Let ρ be a sequence over Q and ρ' be the corresponding sequence of LARs. The set $\mathrm{Inf}(\rho)$ coincides with the hitting set H of the maximum hit h that occurs infinitely often in ρ'.
- The winning condition for the play ρ of a Muller game can be reformulated as follows: *H belongs to $\mathcal{F}$*.
- The winning condition for Muller games can be redefined in terms of a suitable coloring of LAR.
- Parity condition: B wins ρ' if and only if the greatest color that occurs infinitely often in $c(\rho'(0))c(\rho'(1))\ldots$ is even.
- A colored graph (G, c) with the parity condition is said a parity game.

LAR AND PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- The coloring c of LAR , for $h > 0$, can be defined as follows:

$$c(((i_1 \dots i_r), h)) := \begin{cases} 2h & \text{if } \{i_1, \dots, i_h\} \in \mathcal{F}; \\ 2h - 1 & \text{if } \{i_1, \dots, i_h\} \notin \mathcal{F}, \end{cases}$$

where $c(((i_1 \dots i_r), 0)) = 0$

LAR AND PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- The coloring c of LAR , for $h > 0$, can be defined as follows:

$$c(((i_1 \ldots i_r), h)) := \begin{cases} 2h & \text{if } \{i_1, \ldots, i_h\} \in \mathcal{F}; \\ 2h - 1 & \text{if } \{i_1, \ldots, i_h\} \notin \mathcal{F}, \end{cases}$$

  where $c(((i_1 \ldots i_r), 0)) = 0$
- It can be easily shown that the Muller condition $\text{Inf}(\rho) \in \mathcal{F}$ is satisfied if and only if the parity condition is satisfied:
  - $(\Rightarrow)$ if $\text{Inf}(\rho) \in \mathcal{F}$, then $H(= \{i_1, \ldots, i_h\}) \in \mathcal{F}$ and the greatest color that occurs infinitely often is $2h$, which is even.
  - $(\Leftarrow)$ the greatest color that occurs infinitely often is $2h$, which is even, and, thus, the corresponding hitting set belongs to $\mathcal{F}$, from which it follows that $\text{Inf}(\rho) \in \mathcal{F}$.

LAR AND PARITY GAMES

PROOF OF BÜCHI-LANDWEBER THEOREM

- The coloring c of LAR , for $h > 0$, can be defined as follows:

$$c(((i_1 \ldots i_r), h)) := \begin{cases} 2h & \text{if } \{i_1, \ldots, i_h\} \in \mathcal{F}; \\ 2h-1 & \text{if } \{i_1, \ldots, i_h\} \notin \mathcal{F}, \end{cases}$$

  where $c(((i_1 \ldots i_r), 0)) = 0$
- It can be easily shown that the Muller condition $\mathrm{Inf}(\rho) \in \mathcal{F}$ is satisfied if and only if the parity condition is satisfied:
  - ($\Rightarrow$) if $\mathrm{Inf}(\rho) \in \mathcal{F}$, then $H(= \{i_1, \ldots, i_h\}) \in \mathcal{F}$ and the greatest color that occurs infinitely often is $2h$, which is even.
  - ($\Leftarrow$) the greatest color that occurs infinitely often is $2h$, which is even, and, thus, the corresponding hitting set belongs to $\mathcal{F}$, from which it follows that $\mathrm{Inf}(\rho) \in \mathcal{F}$.
- A Muller game $(G, \mathcal{F})$ can be simulated by a parity one (G', c) by means of a finite state machine that transforms a play ρ on G in a corresponding sequence ρ' of LARs (number of LARs $= |Q|! \cdot |Q|$).

PARITY GAMES ARE DETERMINED

PROOF OF BÜCHI-LANDWEBER THEOREM

Let $\text{Attr}_B(S)$ (attractor of B for S) be the set of states from which B can force in a finite number of steps a visit of a state of S.

PARITY GAMES ARE DETERMINED

PROOF OF BÜCHI-LANDWEBER THEOREM

Let $\mathrm{Attr}_B(S)$ (attractor of B for S) be the set of states from which B can force in a finite number of steps a visit of a state of S.

Theorem

A parity game (G, c) is determined and the construction of the winning regions and the positional strategies for A and B are effettive.

PARITY GAMES ARE DETERMINED

PROOF OF BÜCHI-LANDWEBER THEOREM

Let $\mathrm{Attr}_B(S)$ (attractor of B for S) be the set of states from which B can force in a finite number of steps a visit of a state of S.

Theorem

A parity game (G, c) is determined and the construction of the winning regions and the positional strategies for A and B are effettive.

Let $G = (Q, Q_A, E)$, with coloring $c : Q \to \{0, ..., k\}$. We proceed by induction on $|Q|$.

PARITY GAMES ARE DETERMINED

PROOF OF BÜCHI-LANDWEBER THEOREM

Let $\mathrm{Attr}_B(S)$ (attractor of B for S) be the set of states from which B can force in a finite number of steps a visit of a state of S.

Theorem

A parity game (G, c) is determined and the construction of the winning regions and the positional strategies for A and B are effettive.

Let $G = (Q, Q_A, E)$, with coloring $c : Q \to \{0, ..., k\}$. We proceed by induction on $|Q|$.
Base case: trivial.
Inductive step:

- Let the greatest color k be even and let q be a state with color k.

Parity games are determined

Proof of Büchi-Landweber Theorem

Let $\mathrm{Attr}_B(S)$ (attractor of B for S) be the set of states from which B can force in a finite number of steps a visit of a state of S.

Theorem

A parity game (G, c) is determined and the construction of the winning regions and the positional strategies for A and B are effettive.

Let $G = (Q, Q_A, E)$, with coloring $c : Q \to \{0, ..., k\}$. We proceed by induction on $|Q|$.
Base case: trivial.
Inductive step:

- Let the greatest color k be even and let q be a state with color k.
- $A_0 = \mathrm{Attr}_B(\{q\})$. $Q \setminus A_0$ is a subgame.

Parity games are determined

Proof of Büchi-Landweber Theorem

Let $\text{Attr}_B(S)$ (attractor of B for S) be the set of states from which B can force in a finite number of steps a visit of a state of S.

Theorem

A parity game (G, c) is determined and the construction of the winning regions and the positional strategies for A and B are effettive.

Let $G = (Q, Q_A, E)$, with coloring $c : Q \to \{0, ..., k\}$. We proceed by induction on $|Q|$.
Base case: trivial.
Inductive step:

- Let the greatest color k be even and let q be a state with color k.
- $A_0 = \text{Attr}_B(\{q\})$. $Q \setminus A_0$ is a subgame.
- By the inductive hypothesis, we can partition $Q \setminus A_0$ in the two winning regions U_A e U_B for A and B, respectively.

PARITY GAMES ARE DETERMINED (CONT'D)

PROOF OF BÜCHI-LANDWEBER THEOREM

- One of the following two cases necessarily holds:

PARITY GAMES ARE DETERMINED (CONT'D)

PROOF OF BÜCHI-LANDWEBER THEOREM

- One of the following two cases necessarily holds:

  1. From q, player B can force the play to stay in $U_B \cup A_0$ at the next step.

PARITY GAMES ARE DETERMINED (CONT'D)

PROOF OF BÜCHI-LANDWEBER THEOREM

- One of the following two cases necessarily holds:

  1. From q, player B can force the play to stay in $U_B \cup A_0$ at the next step.
     - $W_B = U_B \cup A_0$ and $W_A = U_A$ by applying the positional strategies of the inductive hypothesis on U_A and U_B, the attractor strategy on $\mathrm{Attr}_B(\{q\})$, and the choice of the next state from q according to Case 1.

PARITY GAMES ARE DETERMINED (CONT'D)

PROOF OF BÜCHI-LANDWEBER THEOREM

- One of the following two cases necessarily holds:

1. From q, player B can force the play to stay in $U_B \cup A_0$ at the next step.
   - $W_B = U_B \cup A_0$ and $W_A = U_A$ by applying the positional strategies of the inductive hypothesis on U_A and U_B, the attractor strategy on $\mathrm{Attr}_B(\{q\})$, and the choice of the next state from q according to Case 1.

2. From q, player A can force the play to stay in U_A at the next step.

PARITY GAMES ARE DETERMINED (CONT'D)

PROOF OF BÜCHI-LANDWEBER THEOREM

- One of the following two cases necessarily holds:

1. From q, player B can force the play to stay in $U_B \cup A_0$ at the next step.
   - $W_B = U_B \cup A_0$ and $W_A = U_A$ by applying the positional strategies of the inductive hypothesis on U_A and U_B, the attractor strategy on $\mathrm{Attr}_B(\{q\})$, and the choice of the next state from q according to Case 1.
2. From q, player A can force the play to stay in U_A at the next step.
   - It follows that $q \in \mathrm{Attr}_A(U_A)$. Let us consider now the set $A_1 = \mathrm{Attr}_A(U_A \cup \{q\})$. By applying the inductive hypothesis on the subgame induced by $Q \setminus A_1$, we obtain V_A and V_B. It holds that $W_B = V_B$ e $W_A = V_A \cup A_1$, where the winning positional strategies are given by the inductive hypothesis and the attractor strategy on A_1.

COMPLEXITY ISSUES

PROOF OF BÜCHI-LANDWEBER THEOREM

From the proof of the theorem, it is not difficult to extract a procedure of exponential complexity.

COMPLEXITY ISSUES

PROOF OF BÜCHI-LANDWEBER THEOREM

From the proof of the theorem, it is not difficult to extract a procedure of exponential complexity.

It is known that the problem: "Given a parity game (G, c) and a state q, establish whether or not q belongs to the winning region of B" belongs to the complexity class $\text{NP} \cap \text{co-NP}$.

Complexity issues

Proof of Büchi-Landweber Theorem

From the proof of the theorem, it is not difficult to extract a procedure of exponential complexity.

It is known that the problem: "Given a parity game (G, c) and a state q, establish whether or not q belongs to the winning region of B" belongs to the complexity class NP $\cap$ co-NP.

The possibility of deciding such a problem in polynomial time is one of the most important open problems in the algorithmic theory of infinite games.

COMPLEXITY ISSUES

PROOF OF BÜCHI-LANDWEBER THEOREM

From the proof of the theorem, it is not difficult to extract a procedure of exponential complexity.

It is known that the problem: "Given a parity game (G, c) and a state q, establish whether or not q belongs to the winning region of B" belongs to the complexity class $\mathrm{NP} \cap \text{co-NP}$.

The possibility of deciding such a problem in polynomial time is one of the most important open problems in the algorithmic theory of infinite games.

Remark: equivalence of the above problem and the model checking problem for the μ-calculus.

WHAT NEXT? LTL SYNTHESIS AND BEYOND

A number of variants of Church's problem can be obtained by modifying or generalizing the specification language.

A special attention has been given to the synthesis problem for LTL and other temporal logics.

WHAT NEXT? LTL SYNTHESIS AND BEYOND

A number of variants of Church's problem can be obtained by modifying or generalizing the specification language.

A special attention has been given to the synthesis problem for LTL and other temporal logics.

Manna and Wolper, Synthesis of Communicating Processes from Temporal Logic Specifications, ACM Transactions on Programming Languages and Systems, 6(1): 68-93,1984.

Pnueli and Rosner, On the synthesis problem of a reactive module, In Proc. of POPL, pages 179-190, 1989.

Maler, Pnueli, and Sifakis, On the Synthesis of Discrete Controllers for Timed Systems (An Extended Abstract), In Proc. of STACS, pages 229-242, 1995.

Montanari and Sala, Interval-based Synthesis, In Proc. of GandALF, pages 102-115, 2014.