

# Transforming Game Specifications into Winning Strategies: A General Study

Wladimir Fridman   Jörg Olschewski



GASICS Workshop, 28 June 2009

- 1 Church's Problem and the Büchi-Landweber Theorem
- 2 Regular Winning Conditions
- 3 Context-Free Winning Conditions

# Church's Problem

Problem of Controller Synthesis.



- Input  $\alpha \in \Sigma_1^\omega$       Output  $\beta \in \Sigma_2^\omega$
- $\Sigma = (\Sigma_1 \times \Sigma_2)$        $\begin{pmatrix} \alpha \\ \beta \end{pmatrix} \in \Sigma^\omega$

**Given:** Specification  $L \subseteq \Sigma^\omega$

**Question:** Is there a letter-by-letter transducer, that transforms every input  $\alpha \in \Sigma_1^\omega$  into an output  $\beta \in \Sigma_2^\omega$ , such that  $\begin{pmatrix} \alpha \\ \beta \end{pmatrix} \in L$ ?  
If yes, construct one.

# Infinite Games

- Infinite game with two players.
- They pick letters from their alphabet  $\Sigma_1$  resp.  $\Sigma_2$  in alternation.
- Concatenation of these letter pairs forms an infinite word.

## Example (of a play)

Player 1: a b a a a a a a a a ...

Player 2: b c a a a a a a a a ...

Identify this play with the infinite word  $\gamma = \begin{pmatrix} a \\ b \end{pmatrix} \begin{pmatrix} b \\ c \end{pmatrix} \begin{pmatrix} a \\ a \end{pmatrix} \begin{pmatrix} a \\ a \end{pmatrix} \begin{pmatrix} a \\ a \end{pmatrix} \dots$

- Winning condition: Given by an  $\omega$ -language  $L$  over  $\Sigma$ .
  - If  $\gamma \in L$ , Player 2 wins the play.
  - If  $\gamma \notin L$ , Player 1 wins the play.
- In “Gale-Stewart games” take the interleaving of the two sequences, in example:  $\gamma = abbcaa \dots$

# The Büchi-Landweber Theorem

## Description of Church's Problem

- 1 Given (a finite presentation of)  $L$ , does Player 2 have a winning strategy in the game defined by  $L$ ?
- 2 If yes, compute a winning strategy!

Standard case:  $L$  regular  $\omega$ -language

## Basic Result (Büchi-Landweber 1969)

For a **regular**  $\omega$ -language  $L$ ,

- 1 can be decided, and
- 2 for construction **finite automata** (with output) suffice.

# Connection winning condition – winning strategy

- Usually winning conditions and strategies reside in different domains.
- What does it mean that a strategy is FO-definable?
- We want to establish a connection between winning conditions and winning strategies on the same conceptual level.

# Strategies as Tuples of Languages

- Given a strategy  $f_1$
- For  $c \in \Sigma_1$  introduce

$$K_c = \{w \in \Sigma^* \mid f_1(w) = c\}$$

- Represent  $f_1$  by a tuple of \*-languages  $(K_c)_{c \in \Sigma_1}$
- Similarly for  $f_2$ :

$$K_c = \{w \in (\Sigma)^*(\Sigma_1 \times \{\square\}) \mid f_2(w) = c\}$$

## Example

- Strategy for Player 2:  $(K_a, K_b)$

$$K_a = \Sigma^* \left( \begin{pmatrix} b \\ a \end{pmatrix} \begin{pmatrix} a \\ \square \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} \begin{pmatrix} a \\ \square \end{pmatrix} + \begin{pmatrix} a \\ a \end{pmatrix} \begin{pmatrix} b \\ \square \end{pmatrix} + \begin{pmatrix} b \\ b \end{pmatrix} \begin{pmatrix} b \\ \square \end{pmatrix} \right)$$

$$K_b = \Sigma^* \left( \begin{pmatrix} a \\ a \end{pmatrix} \begin{pmatrix} a \\ \square \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} \begin{pmatrix} b \\ \square \end{pmatrix} + \begin{pmatrix} b \\ b \end{pmatrix} \begin{pmatrix} b \\ \square \end{pmatrix} + \begin{pmatrix} b \\ a \end{pmatrix} \begin{pmatrix} a \\ \square \end{pmatrix} \right)$$

## Theorem (Selivanov 2007, Rabinovich-Thomas 2007)

*“Every  $X$ -game is determined with  $X$  winning strategies.”  
holds for*

- $X = \text{MSO-definable}$
- $X = \text{FO}(<)\text{-definable}$
- $X = \text{FO}(S)\text{-definable}$
- $X = \text{FO}(<)+\text{MOD-definable}$

## But the statement fails for

- $X = \text{FO}(S)+\exists^\omega$
- $X = \text{FO}(S)+\text{MOD}$

## Definition

A  $*$ -language  $L$  is  **$k$ -locally testable**, if membership of  $w$  in  $L$  only depends on

- the set of factors of  $w$  of length  $k$ ,
- the prefix of  $w$  of length  $k - 1$
- and the suffix of  $w$  of length  $k - 1$ .

Refinement:  $k$ -locally  $r$ -threshold testable languages:  
distinguish occurrences by number, up to threshold value  $r$

# Locally Testable $\omega$ -Languages

## Definition

An  $\omega$ -language  $L$  is  **$k$ -locally testable**, if membership of  $\alpha$  in  $L$  only depends on

- the set of factors of  $\alpha$  of length  $k$
- and the prefix of  $\alpha$  of length  $k - 1$ .

In the literature: “finitely locally testable”

## Definition

An  $\omega$ -language is called **strongly locally testable**, if it is of the form

$$L = \bigcup_{i=1}^n U_i V_i^\omega$$

where  $U_i, V_i$  are locally testable.

## Example (A 3-locally testable language $\omega$ -language)

$$L = \{\alpha \in (\{0, 1\}^2)^\omega \mid \text{there is a 1 in the first component or the second component is } (001)^\omega\}$$

- $L$  is 3-locally testable
- Consider typical case that Player 1 always chooses 0
- Player 1: 0 0 0 0 0 0 0 0 0 0 ...
- Player 2: 0 0 1 0 0 1 0 0 1 0 ...
- Player 2 has a 3-locally testable winning strategy:  
if  $\begin{pmatrix} 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ \square \end{pmatrix}$  is a factor of the current play prefix  $\rightsquigarrow 1$  else  $\rightsquigarrow 0$
- There is no 2-locally testable winning strategy:  
when seeing  $\begin{pmatrix} 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ \square \end{pmatrix}$  choose a 0 or a 1?

## Theorem

*Every  $X$ -game is determined with  $X$  winning strategies for the following cases of  $X$ :*

- *locally testable*
- *$k$ -locally  $r$ -threshold testable*
- *piecewise testable*
- *$k$ -piecewise  $r$ -threshold testable*

The statement fails for

$X =$  strongly locally testable  $\omega$ -languages.

# Context-Free Languages

## Definition

A **pushdown automaton** (PDA)  $\mathcal{A} = \langle Q, \Sigma, \Gamma, \delta, q_{in}, \perp \rangle$  consists of

- finite set of states  $Q$ , initial state  $q_{in}$ ,
- input alphabet  $\Sigma$ ,
- stack alphabet  $\Gamma$ , bottom stack symbol  $\perp$ ,
- transition function  $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma^*)$

A PDA  $\mathcal{A}$  is **deterministic**, if  $\forall q, a, Z$  holds  $|\delta(q, a, Z)| + |\delta(q, \varepsilon, Z)| \leq 1$ .

## Definition

**CFL** = class of languages recognized by a PDA with final state set

**CFL <sub>$\omega$</sub>**  = class of  $\omega$ -languages recognized by a PDA with Muller set

**DCFL** and **DCFL <sub>$\omega$</sub>** , the corresponding classes for det. PDAs

# Context-Free Specifications

## Theorem

*Church's Problem for  $L \in CFL_\omega$  is undecidable.*

Reduction:  $UNIVERSALITY(CFL_\omega) \leq CHURCH(CFL_\omega)$

For  $L_1 \in CFL_\omega$  define

$$L := \left\{ \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \in \left( \begin{matrix} \Sigma_1 \\ \Sigma_2 \end{matrix} \right)^\omega \mid \alpha \in L_1, \beta \in \Sigma_2^\omega \right\}$$

Then  $L_1 = \Sigma_1^\omega \iff$  Player 2 has a winning strategy in the game defined by  $L$ . □

## Theorem (Finkel 2001)

*Even for the class of closed  $CFL_\omega$ 's it is undecidable which player has a winning strategy in the Gale-Stewart game defined by  $L$ .*

### Theorem (Walukiewicz 1996)

*Parity games on deterministic pushdown graphs are determined with deterministic pushdown winning strategies.*

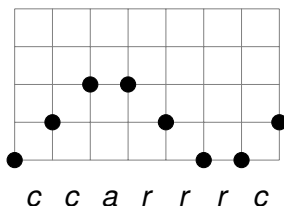
This result can be adapted easily to pushdown winning conditions.

### Theorem

*Games with winning conditions  $L \in DCFL_w$  are determined with winning strategies in DCFL.*

# Visibly Pushdown Languages

- A **VPA** is a PDA with alphabet  $\Sigma = \Sigma_c \cup \Sigma_r \cup \Sigma_{int}$ 
  - call - VPA pushes exactly one symbol:  $\Delta_1 \subseteq Q \times \Sigma_c \times Q \times (\Gamma \setminus \{\perp\})$
  - return - VPA pops one symbol:  $\Delta_2 \subseteq Q \times \Sigma_r \times \Gamma \times Q$
  - internal - stack is preserved:  $\Delta_3 \subseteq Q \times \Sigma_{int} \times Q$



- $\epsilon$ -transitions are not allowed
- defines **VPL** and **VPL <sub>$\omega$</sub>**  in the usual way
- nice closure properties

## Theorem (Löding-Madhusudan-Serre 2004)

*Visibly pushdown games are decidable and a pushdown winning strategy can be computed effectively.*

Developed an equivalent deterministic automata model:

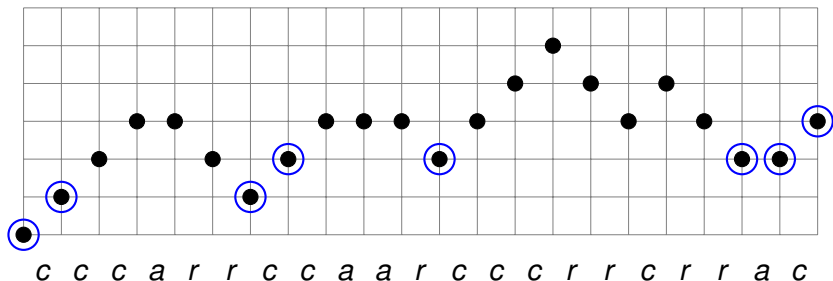
- Stair Visibly Pushdown Automata (**StVPA**)

$$\text{VPL}_\omega = \text{StVPL}_\omega = \text{StDVPL}_\omega$$

## Theorem

*Visibly pushdown games are determined with visibly pushdown winning strategies.*

# Stair Pushdown Automata



$$\text{Steps}_\alpha = \{n \in \mathbb{N} \mid \forall m \geq n: sh(\alpha \upharpoonright m) \geq sh(\alpha \upharpoonright n)\}$$

Evaluation of the acceptance condition only at the steps.

Deterministic PDA with Parity condition  $\rightsquigarrow$  **StDCFL <sub>$\omega$</sub>**

## Theorem

*StDCFL <sub>$\omega$</sub> -games are determined with DCFL winning strategies.*

## Theorem

*Every  $X$ -game is determined with  $X$  winning strategies for the following cases of  $X$ :*

- *DCFL*
- *VPL*
- *StDCFL <sub>$\omega$</sub>  with DCFL winning strategies*
- *real-time-DCFL*

The statement fails for

$X = \text{CFL}$

- Game specifications and strategies can be expressed by (tuples of) languages.
- Locally (threshold) testable games are determined with locally (threshold) testable winning strategies.
- Piecewise (threshold) testable games are determined with piecewise (threshold) testable winning strategies.
- Outlook
  - Consider further language classes, e.g. 1-counter PDAs.
  - Formulate a general result covering the known cases (work in progress).  
Can we find a method to come from  $X$ -winning conditions to  $Y$ -winning strategies?

# Summary

