Jeux pour la théorie des automates, la vérification et l'internet

Olivier Serre and Wiesław Zielonka

LIAFA, Universit Denis Diderot

Lecture 1

Wiesław Zielonka

mail: zielonka@liafa.jussieu.fr

Introduction (appetizer)

Combinatorial games

More about combinatorial games in these books:
Conway, On numbers and games. Berlekamp, Conway, Guy, Winning ways for your mathematical plays.

There is a number of positions, at each position a number of legal moves that lead to a new position. This can be seen as a directed graph with positions=nodes and moves=outgoing arcs. Some positions do not have any legal moves (no outgoing arcs). Players 1 and 2 alternate their moves. The play terminates at a position with no legal moves, the player who should play at such a position loses the game.

The graph is acyclic (no return possible) and no infinite sequence of moves is possible.

Always one of the players has a winning strategy: use backward induction starting from positions without legal moves.

Game Hex

Invented by Pete Hein 1942, reinvented by John Nash.

Hex is played on a hexagonal grid. Two opposite sides of the grid have the same colour, either red or blue, and there are two players that we call also Red and Blue. The players play in turns coloring the hexagons in their respective colors. If there is a red path joining two red sides then Red wins and if there is a blue path connecting two blue sides then Blue wins.

The game can never end in a tie because if all hexagons are colored then either there is a red path connecting the red sides or a blue path connecting the blue sides (this is not so obvious).

The first player has a winning strategy. We can prove it by a non-constructive strategy-stealing argument. To fix the attention suppose that Red is the first player. First of all one of players has a winning strategy since

this is a combinatorial game. Suppose that Blue has a winning strategy S. Then Red can steal Blue's strategy and use it to win himself: at the first turn Red colors any hexagon and next he plays according to S. It can happen that the strategy S indicates to color a hexagon that is already red. In this case Red simply colors any uncolored hexagon (having more red hexagons is never bad for him!)

No explicit winning strategy is known for the first player for grids greater than 9×9 even if we know that such a strategy exists!

Game Nim

k piles of chips.

Positions

$$(n_1, \ldots, n_k)$$

where n_i the number of chips on ith pile. The current player should remove any number of chips (at least one) from exactly one pile. If all piles are empty then the current player has no valid move and he loses.

Winning strategy if one pile is trivial: remove all chips at once.

With two piles: the position (n_1, n_2) is winning for the current player iff $n_1 \neq n_2$.

His strategy: remove chips from the higher pile in order to obtain the configuration with two piles of the same height.

With any number of piles the solution was found by Bouton (1902):

Theorem. *The position* $(n_1, \ldots, n_k)$ *is winning for the current player iff*

$$n_1 \oplus \ldots \oplus n_k \neq 0$$

where $\oplus$ *is the Nim-addition.*

Let $x, y \in \mathbb{N}$ two integers and $x = (x_k, \ldots, x_0)_2$, $y = (y_k, \ldots, y_0)_2$ their binary developments (completed by 0 on the left if necessary). Then

$$x \oplus y = (z_k, \ldots, z_0)_2, \quad \text{where } z_i = x_i \oplus y_i$$

and $x_i \oplus y_i = (x_i + y_i) \mod 2$.

Exercise. Show Bouton's theorem.

Hint: It is sufficient (and necessary) to prove the following two facts:

- If $(n_1, \ldots, n_k)$ is such that $n_1 \oplus \cdots \oplus n_k = 0$ then all legal moves lead to positions $(m_1, \ldots, m_k)$ with the Nim-sum $\neq 0$.

- On the other hand, if $n_1 \oplus \cdots \oplus n_k \neq 0$ then there exists a legal move to a position with the Nim-sum equal to 0.

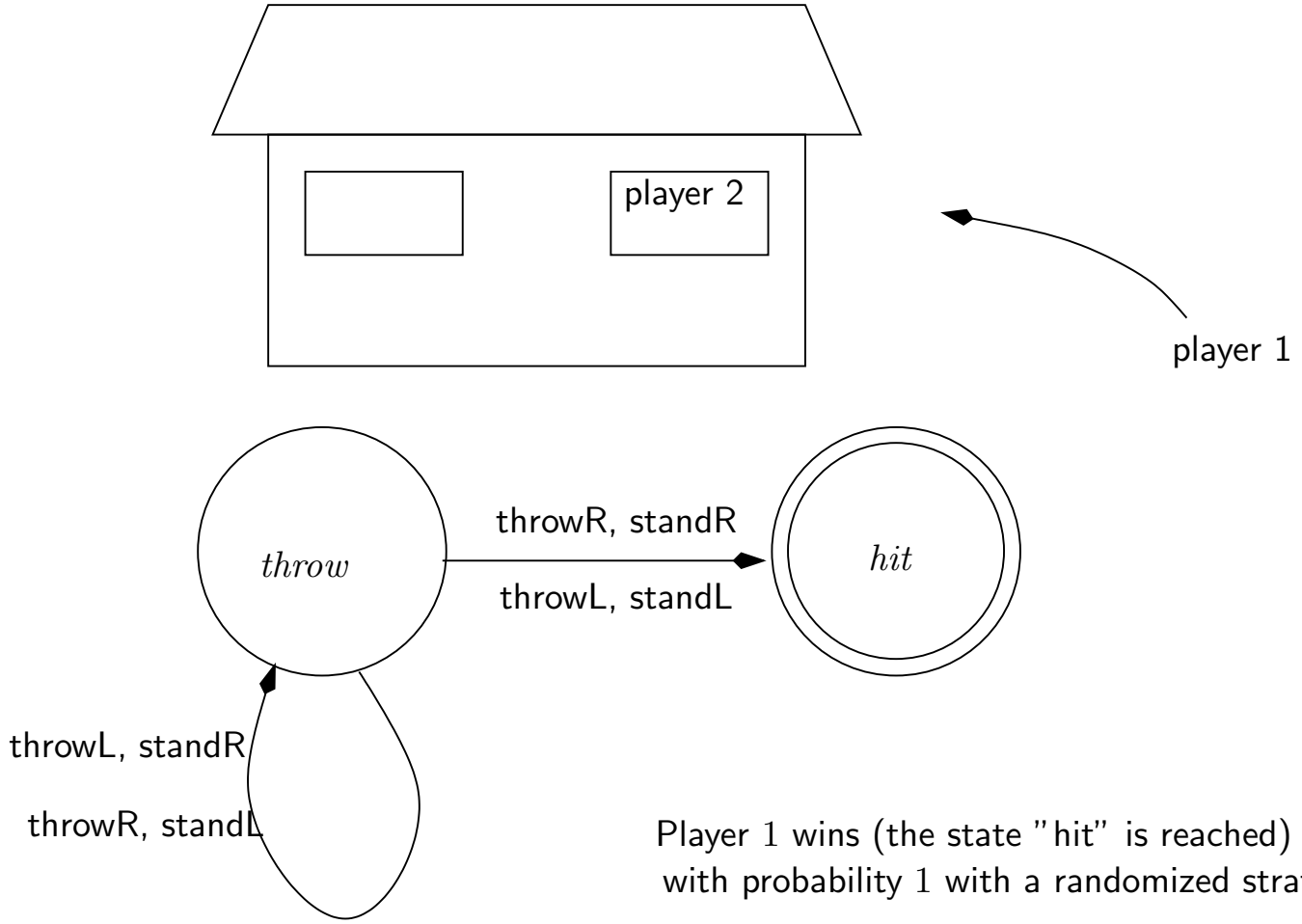
Games for automata and verification

Take a transition system with two disjoint sets of states S_1 and S_2 controlled respectively by player 1 and player 2. Player 1 is, for example, a computer or a program and player 2 the environment or a user, thus we model a system interacting with the environment. For each state there are some available moves and it is the player controlling the current state that chooses the move and this move changes the state of the system. We want that our computer accomplishes its task even if the environment is hostile thus this is a game with two players. Possible winning conditions for player 1:

- visit at least once a given state s,
- more complicated: visit infinitely often the state s,

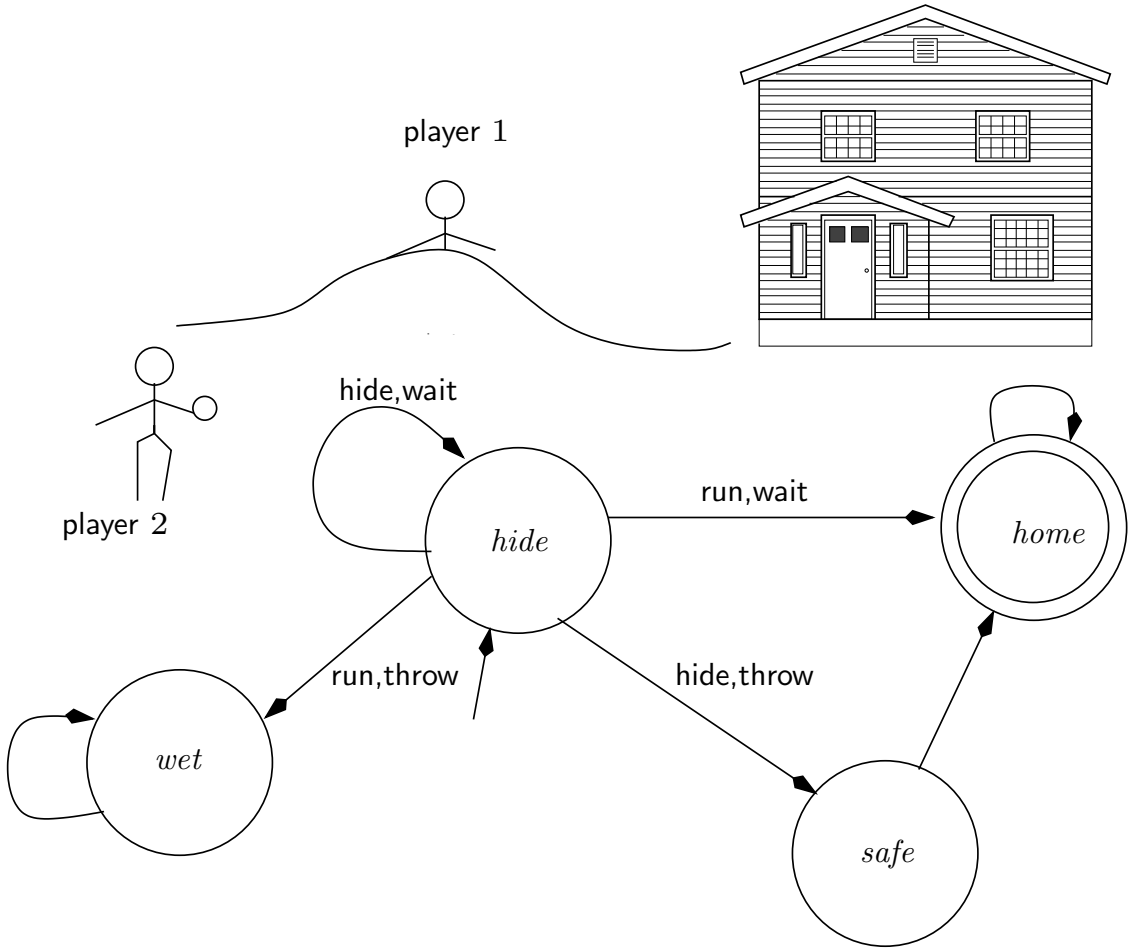
- still more complicated: after every visit to a state x visit, after a finite amount of time, a state y.

Games with simultaneous moves (imperfect information)



Player 1 wins (the state "hit" is reached)
with probability 1 with a randomized strategy.

ϵ-optimal but not optimal strategy (Game Hide or Run)



If player 1 chooses **run** with probability ϵ at each round than he will run at some moment with probability 1 but for any strategy of player 2 the

In the game Hide or Run any finite memory strategy of player 2 either

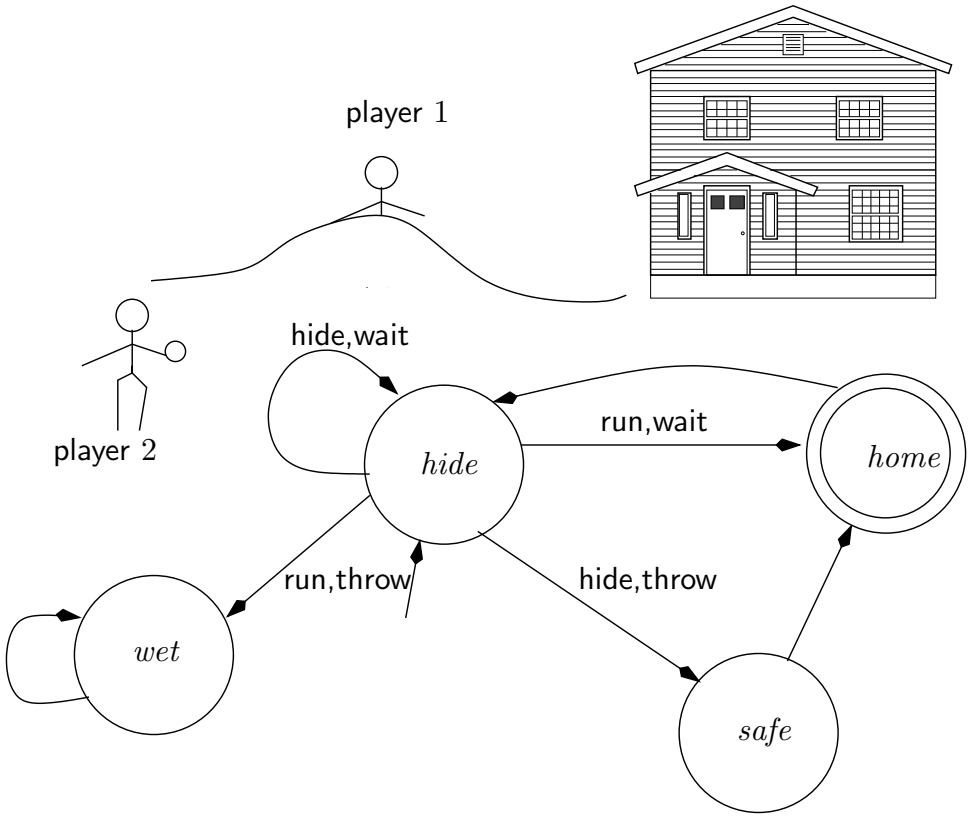
- never throws after some finite amount of time (and then player 1 can run to home) or

- the probability that he never throws is 0 and then, waiting sufficiently long, player 1 can run safely.

The good behavioral strategy of player 2 is

$$\tau(hide^k)(wait) = 2^{-1/2^k}$$

Repeated Hide and Run



In repeated Hide or Run game player 1 returns behind the hill after each successful visit in the shelter. The transition diagram above indicates that, intuitively, if player 2 has thrown the snowball without hitting player 1 then he cannot immediately prepare and use another snowball, he should wait

Player 1 wins if and only if he visits the shelter infinitely often without being hit.

The value of the repeated Hide and Hit game for player 1 at state $hide$ is still 1.

But now he needs an infinite memory to win with probability $> 1 - \epsilon$.

Let

$$\epsilon_k = 1 - (1 - \epsilon)^{-1/2^{k+1}}.$$

Then

$$\prod_{k=0}^{\infty}(1 - \epsilon_k) = 1 - \epsilon$$

His strategy: at **hide** choose **run** with probability ϵ_k where k is the number of prior visits to **home**.

Games and Internet

Routing for selfish users

Each user controls only a very small (negligible) part of the traffic.

Each user chooses a path from a source vertex to a destination vertex. For each edge there is a latency (delay, cost) that depends on the amount of traffic over this edge. The user's objective is to minimize his total latency and his behaviour is selfish, he does not care about the general welfare.

Pigou's example

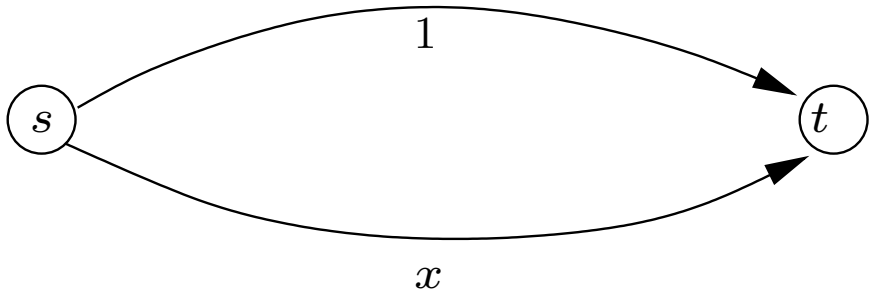


Figure 1: The latency of upper edge is $\ell(x) = 1$, the latency of lower edge is $\ell(x) = x$. The traffic rate between s and t is 1.

Braess's paradox

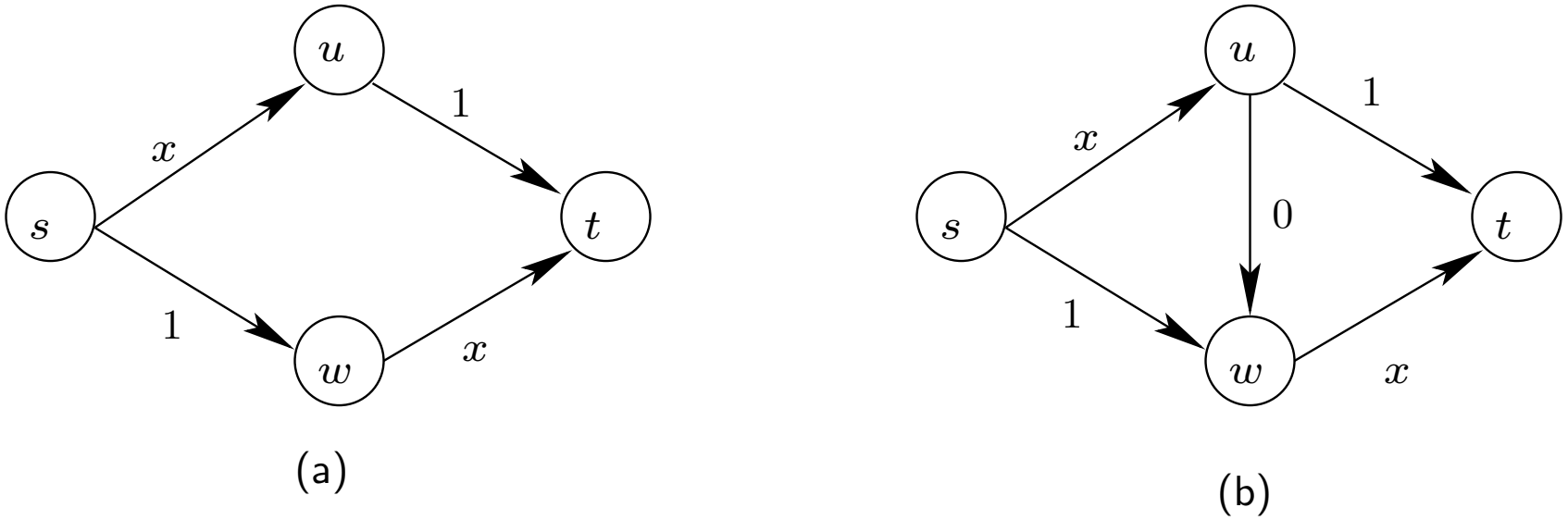


Figure 2: The traffic rate is 1. Latency functions are the same as in the previous example. In (b) the latency on (u, w) edge is 0.

Mechanism design

Instead of solving games we can try to construct games in such a way that selfish and self-interested players will act in a way that optimizes the criteria imposed by the game designer.

For the routing problem : let the users pay for their traffic, or artificially increase the latency.

In other words we should design a game instead of solving it.

Shortest path

A communication network represented by a directed graph. Two special vertices x and y (source and destination). Each edge is controlled by an agent and when a message is send through an edge e it incurs a cost c_e for the controlling agent. The cost is know to the agent but to nobody else.

The goal is to find the cheapest path from x to y.

Problem: find a payment mechanism such that the best strategy for each agent is to announce his/her real cost, i.e. the payment mechanism such that it is useless to lie!