Infinite games on finite graphs

Madhavan Mukund

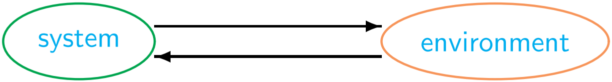
Chennai Mathematical Institute
http://www.cmi.ac.in/~madhavan

Formal Methods Update 2006, IIT Guwahati
3 July 2006

Reactive systems

- Traditionally, computer programs are transformational
  Compute output as a function of inputs

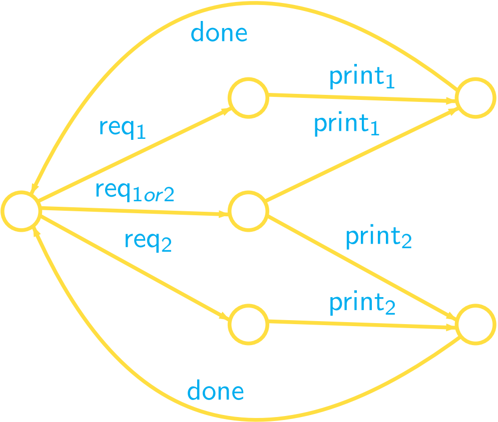
- Inadequate to describe schedulers, operating systems ...
  Reactive systems



- Describe continuous interaction between system and environment as an infinite game

Modelling reactive systems

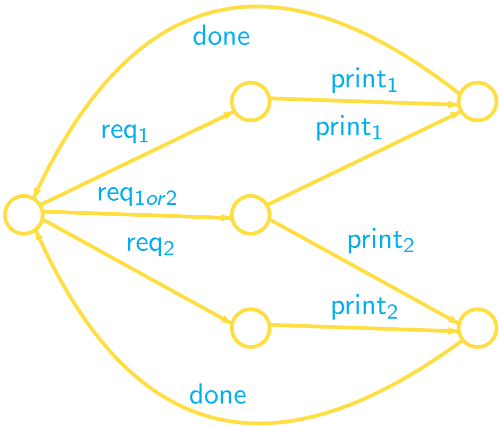
A scheduler that allocates requests to two printers



Computation is a sequence of actions , typically infinite

req$_1$ print$_1$ done req$_{1or2}$ print$_1$ done req$_{1or2}$ print$_2$ done . . .

Desirable and undesirable computations



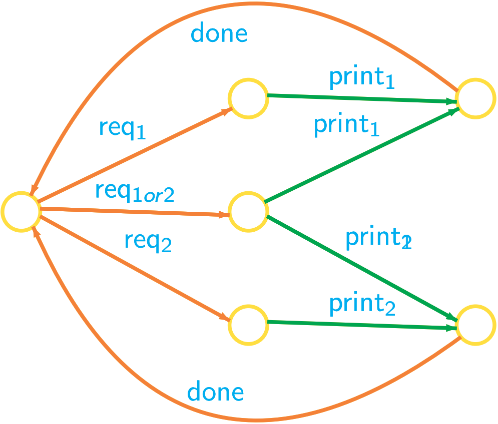
Printer 1 is colour printer, Printer 2 is black and white

Schedule jobs to minimize cost — respond to req$_{1or2}$ with print$_2$

req$_1$ print$_1$ done req$_{1or2}$ print$_1$ done req$_{1or2}$ print$_2$ done ... is bad

req$_1$ print$_1$ done req$_{1or2}$ print$_2$ done req$_{1or2}$ print$_2$ done ... is OK

Controllable and uncontrollable actions



Requests are uncontrollable, choice of printer is controllable

Select controllable actions to achieve objective

— Respond to req$_{1or2}$ with print$_2$

Controllability

- Given a system and an objective, is there a strategy to select controllable actions such that the objective is realized?
- Can this strategy be effectively computed?

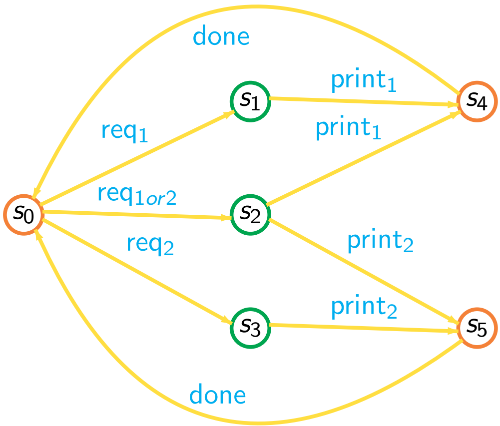
Controllability . . . as a game

- Given a system and an objective, is there a strategy to select controllable actions such that the objective is realized?
- Can this strategy be effectively computed?
- Formulate the problem as a game
  - Two players, system and environment
  - Can select moves for system
  - Control objective is represented as the winning criterion for the game
  - Controllability is a winning strategy for system

Infinite games on finite graphs

- Two players, Player 0 and Player 1
- Moves are determined by a finite game graph with positions labelled 0 or 1.
  - Assume neither player ever gets stuck
  - Moves need not be strictly alternating
- A play of the game is an infinite path through the graph
- Winning condition
  - Some infinite sequences of states are good
  - Player 0 wins if the path chosen is describes a good sequence
  - Otherwise Player 1 wins

Infinite games on finite graphs . . .



- Player 0 plays at green positions, Player 1 at orange positions
- Winning condition: every s_2 is immediately followed by s_5

Winning conditions

- How are the winning conditions specified?
- Simplest winning condition is reachability
  - A set $\mathbf{G}$ of good states
  - Want to visit some state in $\mathbf{G}$ at least once
- Working backwards, compute $\mathbf{Reach(G)}$, the set of states from which Player 0 can force the game to visit $\mathbf{G}$
- Compute $\mathbf{Reach(G)}$ iteratively
- $\mathbf{R_0 = G}$ — if already in $\mathbf{G}$, we have visited $\mathbf{G}$
- $\mathbf{R_{i+1}}$: states from which Player 0 can force game into $\mathbf{R_i}$
  - 0 plays at $\mathbf{s}$, some move from $\mathbf{s}$ to $\mathbf{s' \in R_i} \Rightarrow$ add $\mathbf{s}$ to $\mathbf{R_{i+1}}$
  - 1 plays at $\mathbf{s}$, every move from $\mathbf{s}$ leads to $\mathbf{s' \in R_i} \Rightarrow$ add $\mathbf{s}$ to $\mathbf{R_{i+1}}$
- Eventually $\mathbf{R_{i+1} = R_i}$ because set of states is finite
- This is $\mathbf{Reach(G)}$

Winning conditions — recurrence (Büchi condition)

- Want to visit a set **G** of good states infinitely often
- Reach some $\mathbf{g} \in \mathbf{G}$, such that from **g** we can return to **g** as many times as we want
  - Must leave **g** and then get back
- $\mathbf{Reach^{+}(G)}$: states from which we can reach **G** in one or move moves
  - **Reach(G)** : states from which **G** is reachable in zero or move moves
- Calculate $\mathbf{Reach^{+}(G)}$ iteratively, like **Reach(G)**
- $\mathbf{R_0^{+}}$ is set of states from we can reach **G** in one move
  - When computing **Reach(G)**, $\mathbf{R_0 = G}$
- $\mathbf{R_{i+1}^{+}}$: states where Player 0 can force game into $\mathbf{R_i^{+}}$, as before
- Eventually $\mathbf{R_{i+1}^{+} = R_i^{+} = Reach^{+}(G)}$

Winning conditions — recurrence (Büchi) . . .

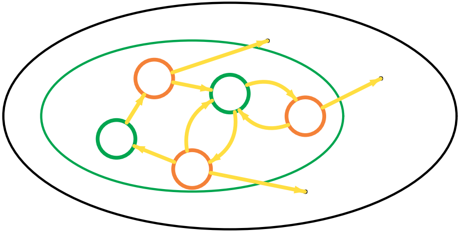
- Want to visit a set **G** of good states infinitely often
- **Reach$^+$(G) $\cap$ G** — states in **G** from which we can return to **G** once
- **Reach$^+$(Reach$^+$(G) $\cap$ G) $\cap$ G** — states in **G** from which we can return to **G** twice
- . . .
- Converges to **Recur(G)** — states in **G** from which we can return to **G** infinitely often
- **Reach(Recur(G))** is the set of states from which Player 0 can start and win the game

Strategies and memory

- For reachability game, Player 0 wins from state s if $s \in \mathbf{Reach(G)}$
  - $s \in R_i$ for some R_i when computing $\mathbf{Reach(G)}$
  - Call this the rank of s
  - s has at least one successor of lower rank : uniformly fix one and choose it every time we are at s
- Strategy "decrease rank" depends only on s — no memory is required
- Recurrence game also has memoryless strategy
  - Initially play decrease rank till we reach $\mathbf{Recur(G)}$
  - Every Player 0 state $s \in \mathbf{Recur(G)}$ is in $\mathbf{Reach^+(Recur(G))}$: again play decrease rank to revisit $\mathbf{Recur(G)}$

Determinacy

- What happens outside **Reach(Recur(G))**?
- Trap for Player 0 : set of states **X** such that
  - For Player 0, all moves from **X** lead back to **X**
  - For Player 1, at least one move from **X** leads back to **X**

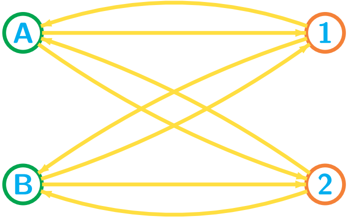


- Player 0 cannot leave the trap and Player 1 can force Player 0 to stay in the trap

Determinacy . . .

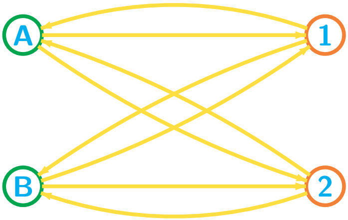
- Complement of **Reach(Recur(G))** is a 0 trap
  - In general, for any set **X**, the complement of **Reach(X)** is a 0 trap
- If the game starts outside **Reach(Recur(G)**, Player 1 can keep the game outside **Reach(Recur(G)** and win
- Büchi games are determined
  From every position, either Player 0 wins or Player 1 wins
- This is a special case of a very general result for infinite games [Martin, 1975]

More complicated winning conditions



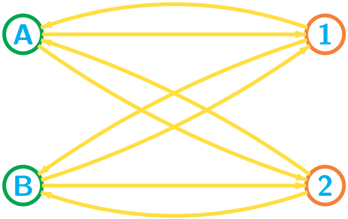
- A play in this game is a sequence in which states $\{1,2\}$ alternate with $\{\mathbf{A},\mathbf{B}\}$
- Player 0 wins if the highest number that appears infinitely often is equal to the number of letters that appear infinitely often
  - If only **A** or **B** appear infinitely often, **2** should not appear infinitely often
  - If both **A** and **B** appear infinitely often, **2** should appear infinitely often

More complicated winning conditions . . .



- A memoryless strategy will force Player 0 to uniformly respond with a move to **1** or **2** from **A** and from **B**
  - If Player 0 chooses **1** from both, Player 1 alternates **A** and **B**
  - If Player 0 chooses **1** from **A** and **2** from **B**, Player 1 always plays **B**
  - If Player 0 chooses **2** from **A** and **1** from **B**, Player 1 always plays **A**
  - If Player 0 chooses **2** from both, Player 1 uniformly chooses **A** (or **B**)

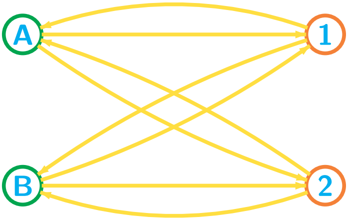
More complicated winning conditions . . .



- Player 0 should remember what Player 1 has played
  - Choose **1** if the latest move by Player 1 is the same as the previous move
  - Choose **2** if the latest move by Player 1 is different from the previous move
- This is a finite memory strategy — Player 0 only needs to remember one previous move of Player 1

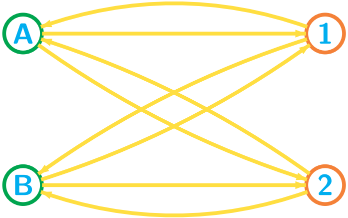
More complicated winning conditions . . .

- Muller condition: family of good sets $(\mathbf{G_1}, \mathbf{G_2}, \ldots, \mathbf{G_k})$
  Set of states visited infinitely often should exactly be one of the $\mathbf{G_i}$'s
- The winning condition of the previous example can be represented as the family
  $(\{1, A\}, \{1, B\}, \{2, A, B\}, \{1, 2, A, B\})$



Strategies and memory

- Need a systematic way to maintain bounded history
- Later Appearance Record (LAR)
  - Remember relative order of last visit to each state
  - Hit position, where last change occurred



- **A → A1 → A1B → A1B2 → •1B2A → 1B • A2 → 1 • A2B → 1A • B2 → 1 • B2A → 1B • A2 → 1 • A2B → 1A • B2 → 1 • B2A → ⋯**

Analyzing LAR

- States visited only finite number of times eventually stay to left of hit position
- If exactly $s_1, s_2, \ldots, s_n$ are visited infinitely often, then infinitely often the LAR will be of the form $\alpha \bullet \beta$ where, among the states visited so far,
  - α is the set of states visited finite number of times
  - β is a permutation of $s_1, s_2, \ldots, s_n$
- Consider a run $A \rightarrow 1 \rightarrow B \rightarrow 2 \rightarrow A \rightarrow 2 \rightarrow B \rightarrow 2 \rightarrow A \rightarrow 2 \rightarrow \cdots$, visiting $\{A, B, 2\}$ infinitely often
- LAR evolves as
  $A \rightarrow A1 \rightarrow A1B \rightarrow A1B2 \rightarrow \bullet 1B2A \rightarrow 1B \bullet A2$
  $\rightarrow 1 \bullet A2B \rightarrow 1A \bullet B2 \rightarrow 1 \bullet B2A \rightarrow 1B \bullet A2$
  $\rightarrow 1 \bullet A2B \rightarrow 1A \bullet B2 \rightarrow 1 \bullet B2A \rightarrow \cdots$

A new winning condition

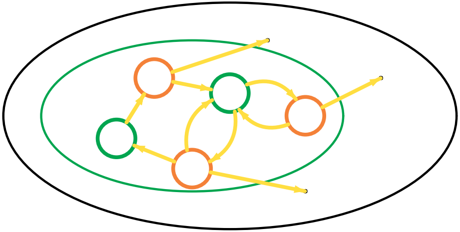
- Muller condition $(G_1, G_2, \ldots, G_k)$
- Expand state space to include LAR: states are now (s, ℓ)
- E_i : (s, ℓ) s.t. $\ell = \alpha \bullet \beta$ an LAR with hit position $< i$
- F_i : E_i plus (s, ℓ) s.t. $\ell = \alpha \bullet \beta$ an LAR with hit position $= i$ and β a permutation of some Muller set G_j
- $E_1 \subsetneq F_1 \subsetneq E_2 \subsetneq \cdots \subsetneq E_n \subsetneq F_n$
  - Merge (E_i, E_{i+1}) if $F_i \setminus E_i = \emptyset$
  - Merge (F_i, F_{i+1}) if $E_{i+1} \setminus F_i = \emptyset$
- Among $E_1 \subsetneq F_1 \subsetneq \cdots \subsetneq E_n \subsetneq F_n$, consider largest set that appears infinitely often
  - If this set is some E_i, Player 0 loses
  - If this set is some F_i, Player 0 wins
- Rabin chain condition

Parity condition

- Rabin chain condition $\mathbf{E_1} \subsetneq \mathbf{F_1} \subsetneq \cdots \subsetneq \mathbf{E_n} \subsetneq \mathbf{F_n}$
- Player 0 wins if "index" of largest infinitely occurring set is even
- Colour states with colours $\{1, 2, \ldots, 2n\}$
  - States in $\mathbf{E_1}$ get colour $\mathbf{1}$
  - States in $\mathbf{F_1} \setminus \mathbf{E_1}$ get colour $\mathbf{2}$
  - ...
  - States in $\mathbf{E_i} \setminus \mathbf{F_{i-1}}$ get colour $\mathbf{2i-1}$
  - States in $\mathbf{F_i} \setminus \mathbf{E_i}$ get colour $\mathbf{2i}$
- Player 0 wins if largest colour visited infinitely often is even
- Parity condition

Parity games have memoryless winning strategies

- Trap for Player 0 : set of states **X** such that
  - For Player 0, all moves from **X** lead back to **X**
  - For Player 1, at least one move from **X** leads back to **X**
  - Player 0 cannot leave the trap and Player 1 can force Player 0 to stay in the trap



- Trap for Player 1 : symmetric
- For any **X**, **S** $\setminus$ **Reach(X)** is a 0 trap

Parity games have memoryless winning strategies . . .

- A set of positions $\mathbf{U}$ is a 0-paradise if $\mathbf{U}$ is a 1 trap in which Player 0 has a winning strategy
- Define a 1-paradise symmetrically

Theorem

The set of positions of a parity game can be partitioned into a 0-paradise and a 1-paradise

- Proof is by induction on the size of largest colour $\mathbf{n}$ used to label positions
- Base case: $\mathbf{n = 0}$
  - Only Player 0 can win
  - Entire set of positions is a 0 paradise

Parity games have memoryless winning strategies . . .

- Assume $\mathbf{n} > \mathbf{0}$ is even ($\mathbf{n}$ odd is symmetric)

- Suppose X_1 is an 1-paradise and complement X_0 is a 1 trap

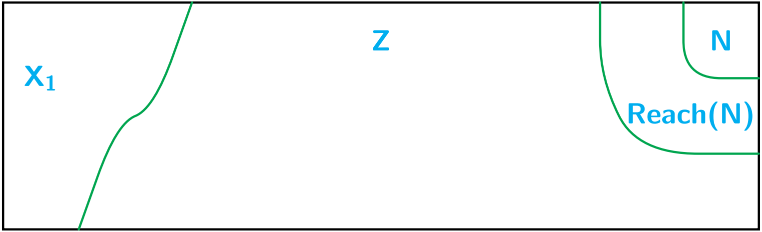
Parity games have memoryless winning strategies . . .

- Assume $n > 0$ is even (n odd is symmetric)

- Suppose X_1 is an 1-paradise and complement X_0 is a 1 trap
- Let $N \subseteq X_O$ be states with colour n

Parity games have memoryless winning strategies ...

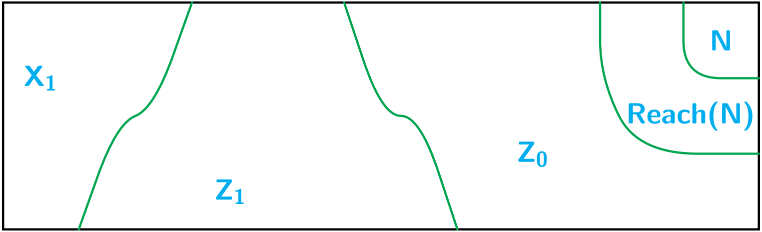
- Assume $n > 0$ is even (n odd is symmetric)



- Suppose X_1 is an 1-paradise and complement X_0 is a 1 trap
- Let $N \subseteq X_O$ be states with colour n
- Let Z be $X_0 \setminus$ Reach(N)

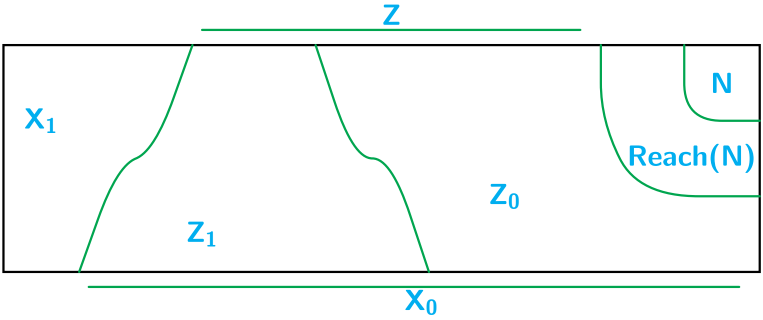
Parity games have memoryless winning strategies . . .

- Assume $\mathbf{n} > 0$ is even ($\mathbf{n}$ odd is symmetric)



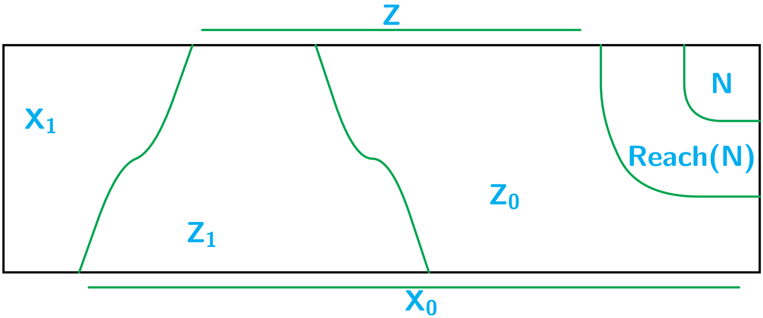
- Suppose $\mathbf{X_1}$ is an 1-paradise and complement $\mathbf{X_0}$ is a 1 trap
- Let $\mathbf{N} \subseteq \mathbf{X_O}$ be states with colour $\mathbf{n}$
- Let $\mathbf{Z}$ be $\mathbf{X_0} \setminus \mathbf{Reach(N)}$
- $\mathbf{Z}$ is a subgame with parities $< \mathbf{n}$
  Inductively, split $\mathbf{Z}$ as 1 paradise $\mathbf{Z_1}$ and 0 paradise $\mathbf{Z_0}$

Parity games have memoryless winning strategies . . .



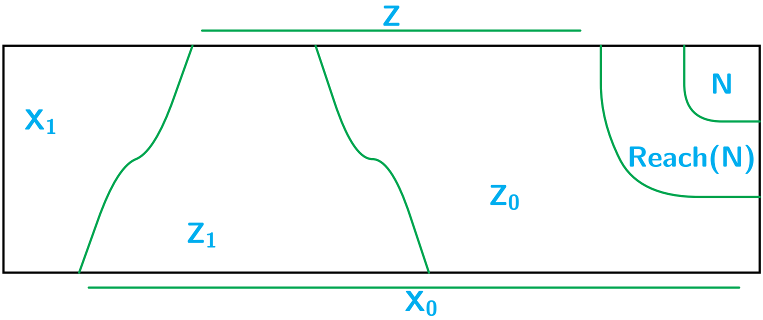
- If Z_1 is nonempty, we can extend 1 paradise X_1 to $X_1 \cup Z_1$
  - Z is a 0 trap in X_0, Z_1 is a 0 trap in $Z \Rightarrow Z_1$ is a 0 trap in X_0
  - $X_1 \cup Z_1$ is a 0 trap
  - If game stays in Z_1, 1 wins Z game
  - If game moves to X_1, 1 wins in X_1

Parity games have memoryless winning strategies . . .



- If Z_1 is nonempty, we can extend 1 paradise X_1 to $X_1 \cup Z_1$
- If Z_1 is empty, X_0 is a 0 paradise
  - From N, return to X_0
  - From Reach(N) return to N
  - From Z_0 win Z_0 game

Parity games have memoryless winning strategies . . .



- If Z_1 is nonempty, we can extend 1 paradise X_1 to $X_1 \cup Z_1$
- If Z_1 is empty, X_0 is a 0 paradise
- Recursively partition positions into 0 and 1 paradise, starting with X_1 empty

Concluding remarks

- Problem originally posed by Church/Büchi, solved by Büchi and Landweber in 1969
- Can be extended to certain kinds of infinite game graphs that are finitely generated
  - Pushdown graphs, corresponding to an automaton with a stack
- The model checking problem for modal μ-calculus directly reduces to solving parity games
- What is the complexity of constructing a memoryless winning strategy for parity games?
  - Our recursive algorithm has complexity $O(mn^d)$ for a game with m edges, n positions, d colours
  - The problem is in $NP \cap co(NP)$. Is it in P?
- Can we do improve on LAR for winning conditions that require memory?

References

- Robert McNaughton:
  Infinite games played on finite graphs,
  Annals of Pure and Applied Logic, **65** (1993) 149–184.
- Wolfgang Thomas:
  On the synthesis of strategies in infinite games,
  Proceedings of STACS 1995,
  Springer Lecture Notes in Computer Science (1995).
- Erich Grädel, Wolfgang Thomas, Thomas Wilke (Eds.):
  Automata, Logics, and Infinite Games: A Guide to Current Research,
  Springer Lecture Notes in Computer Science Vol 2500 (2002).