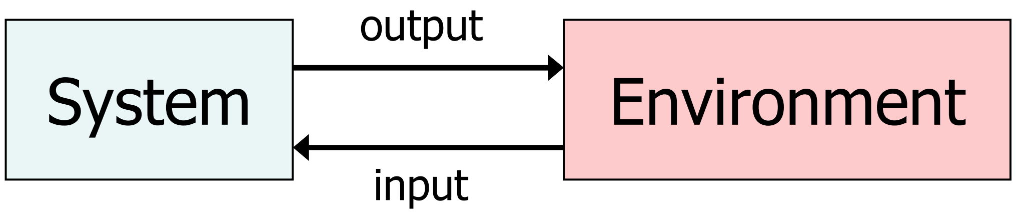
Energy and Mean-Payoff Parity Markov Decision Processes

Laurent Doyen
LSV, ENS Cachan & CNRS

Krishnendu Chatterjee
IST Austria

MFCS 2011

Games for system analysis

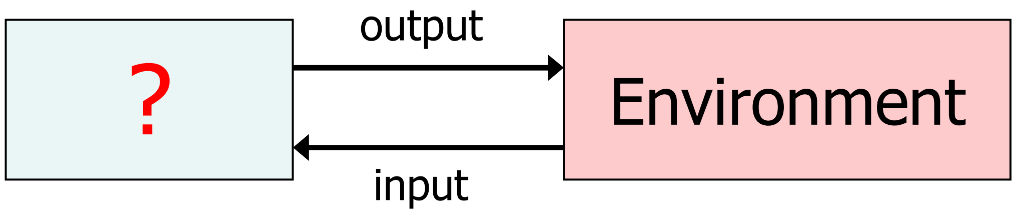


Spec: φ(input,output)

- Verification: check if a given system is correct

  → reduces to graph searching

Games for system analysis



Spec: φ(input,output)

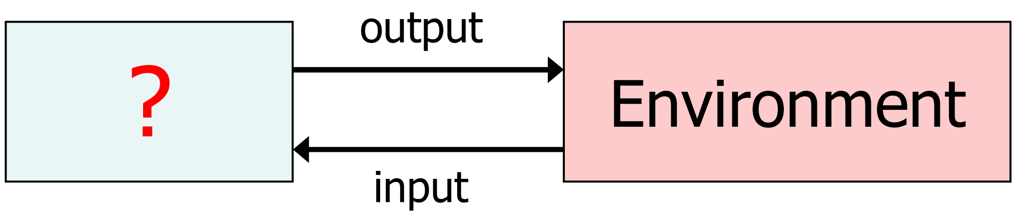
- Verification: check if a given system is correct

  → reduces to graph searching

- Synthesis : construct a correct system

  → reduces to game solving – finding a winning strategy

Games for system analysis



Spec: φ(input,output)

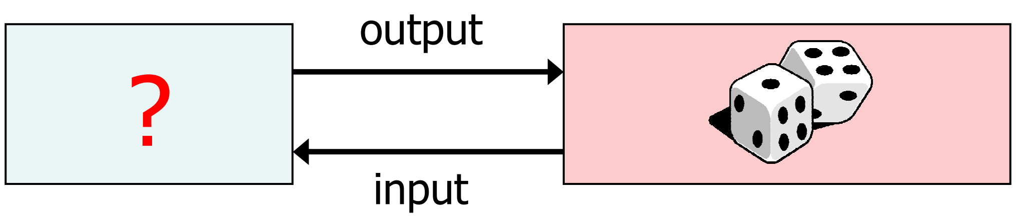
- Verification: check if a given system is correct

  → reduces to graph searching

- Synthesis : construct a correct system

  → reduces to game solving – finding a winning strategy

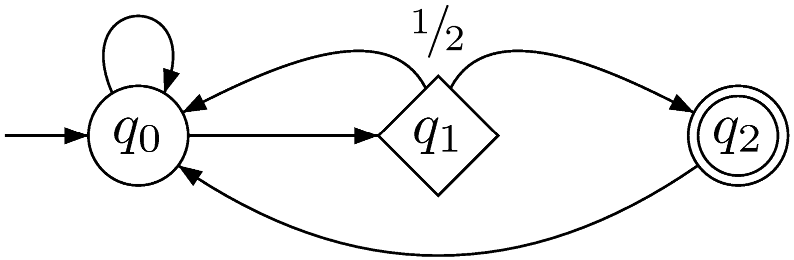
This talk: environment is abstracted as a stochastic process



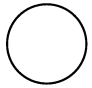
= Markov decision process (MDP)

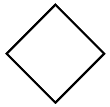
Markov decision process

Markov decision process (MDP)

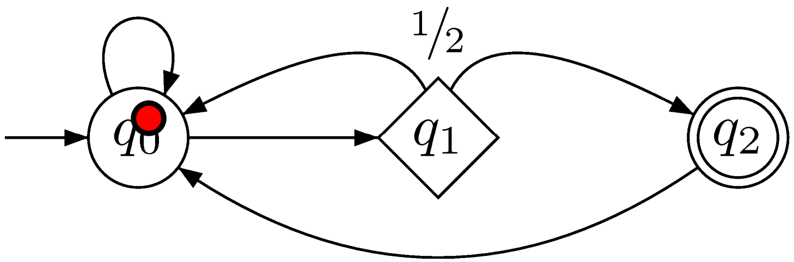


$$Q = Q_\circ \cup Q_\diamond$$

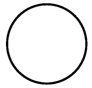
 Nondeterministic (player 1)

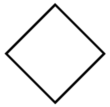
 Probabilistic

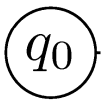
Markov decision process (MDP)



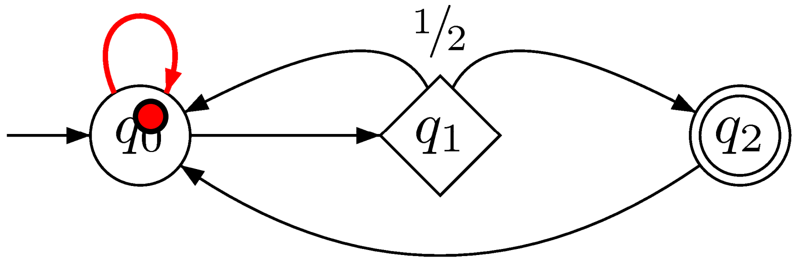
$$Q = Q_\circ \cup Q_\diamond$$


Nondeterministic (player 1)


Probabilistic



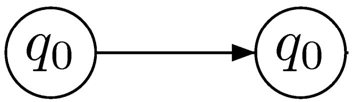
Markov decision process (MDP)



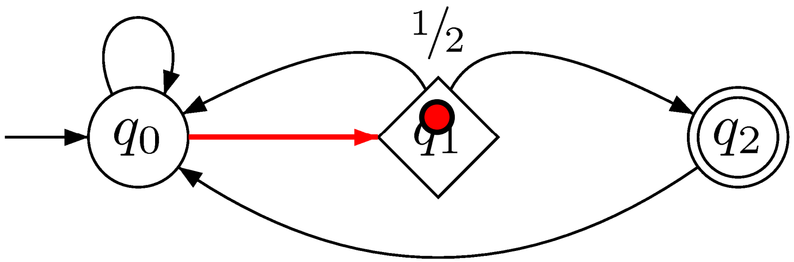
$$Q = Q_\circ \cup Q_\diamond$$

○ Nondeterministic (player 1)

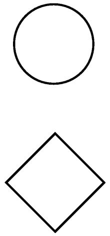
◇ Probabilistic

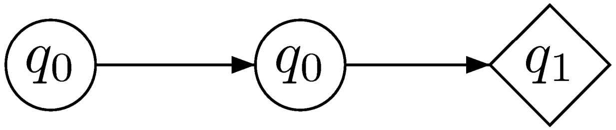


Markov decision process (MDP)

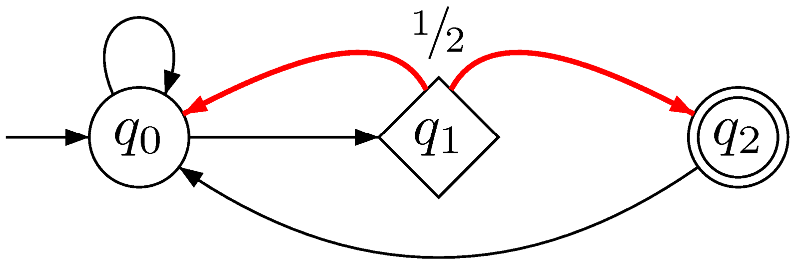


$$Q = Q_\circ \cup Q_\diamond$$

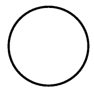


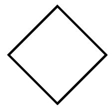


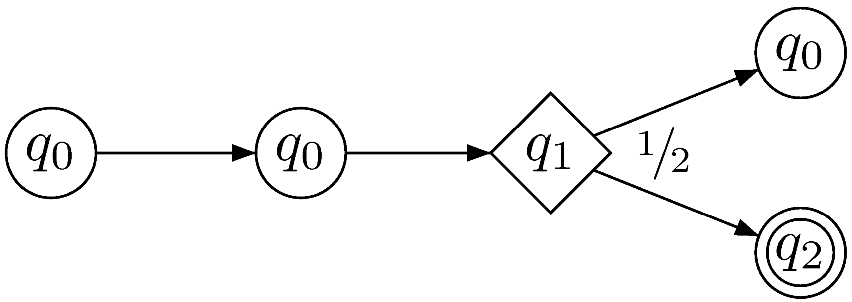
Markov decision process (MDP)



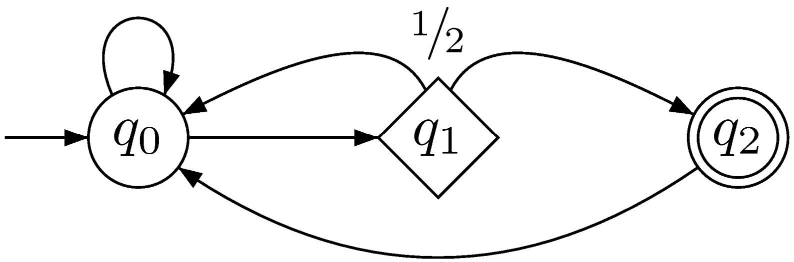
$$Q = Q_\circ \cup Q_\diamond$$

 Nondeterministic (player 1)

 Probabilistic



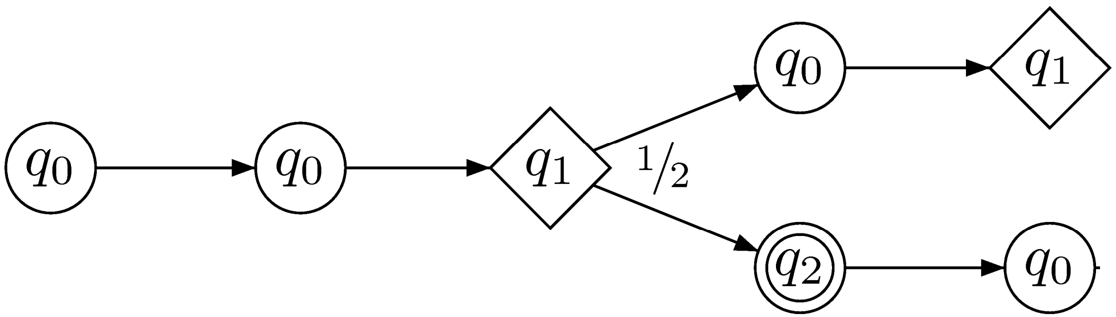
Markov decision process (MDP)



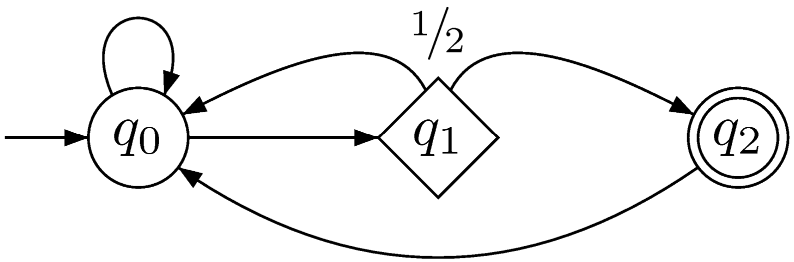
$$Q = Q_\circ \cup Q_\diamond$$

○ Nondeterministic (player 1)

◇ Probabilistic



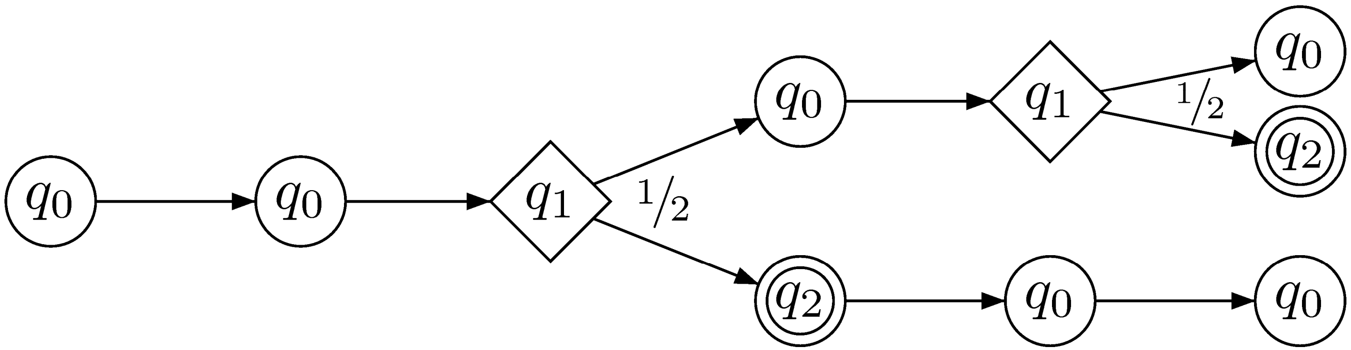
Markov decision process (MDP)



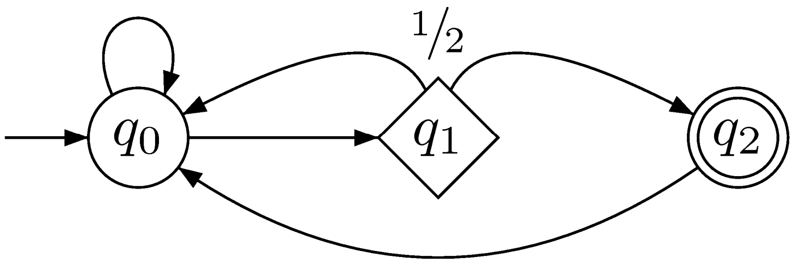
$$Q = Q_\circ \cup Q_\diamond$$

○ Nondeterministic (player 1)

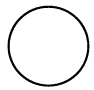
◇ Probabilistic



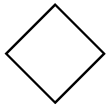
Markov decision process (MDP)



$Q = Q_\circ \cup Q_\diamond$



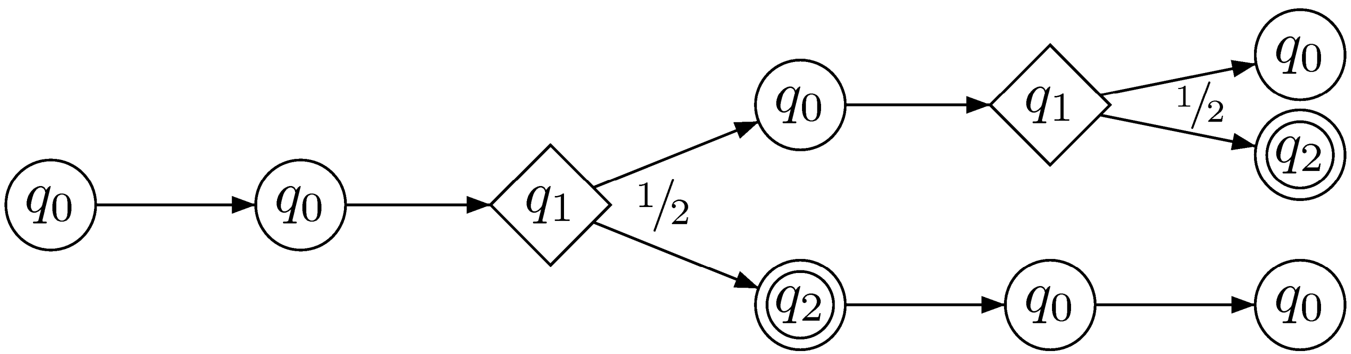
Nondeterministic (player 1)



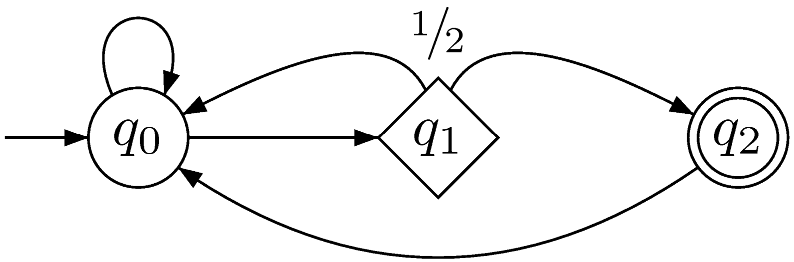
Probabilistic

Strategy (policy) = recipe to extend the play prefix

$\sigma : Q^* \cdot Q_\circ \to Q$



Markov decision process (MDP)



$Q = Q_\circ \cup Q_\diamond$

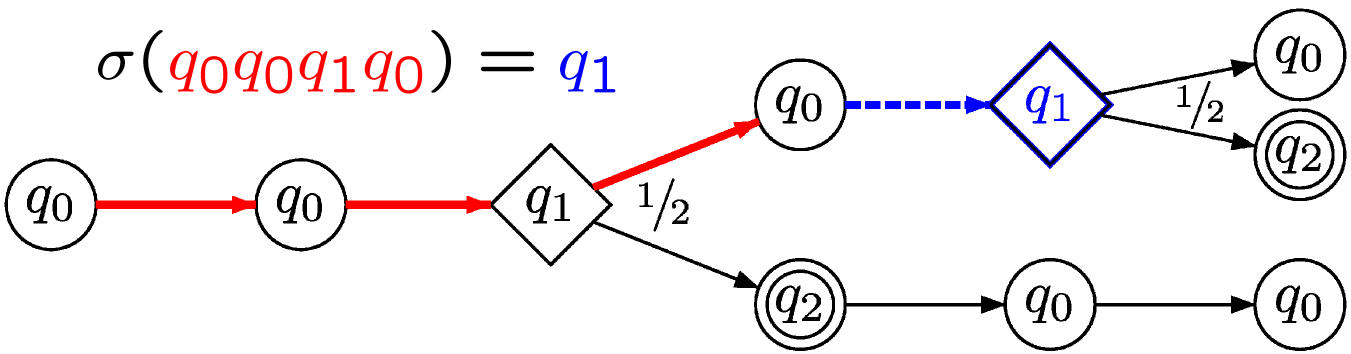
○ Nondeterministic (player 1)

◇ Probabilistic

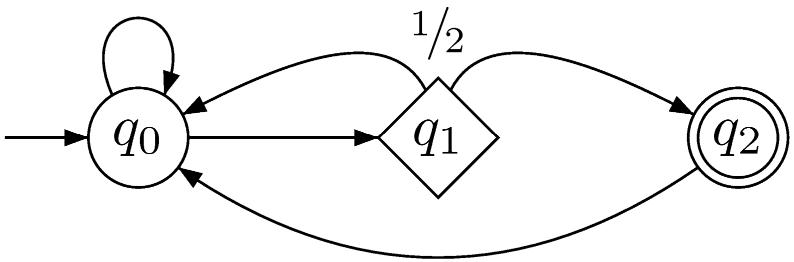
Strategy (policy) = recipe to extend the play prefix

$\sigma : Q^* \cdot Q_\circ \to Q$

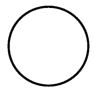
$\sigma(q_0 q_0 q_1 q_0) = q_1$

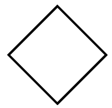


Markov decision process (MDP)



$$Q = Q_\circ \cup Q_\diamond$$

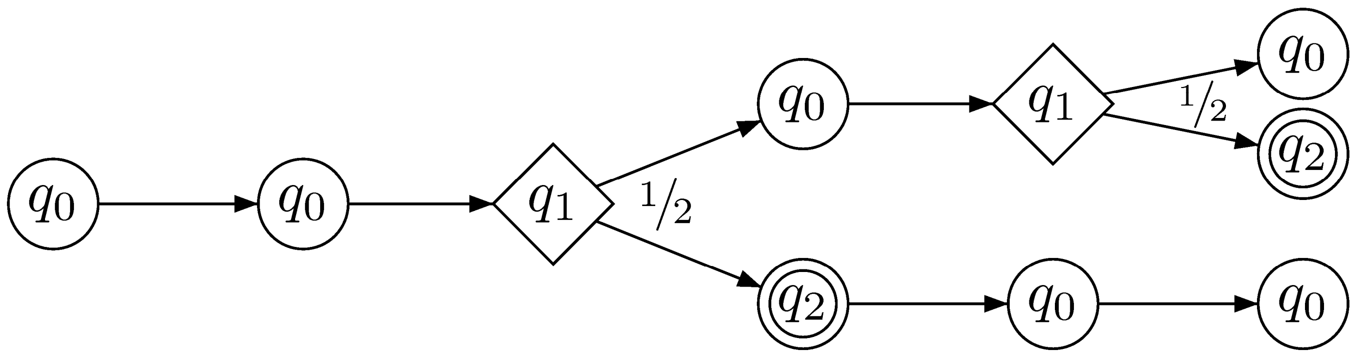
 Nondeterministic (player 1)

 Probabilistic (player 2)

weak

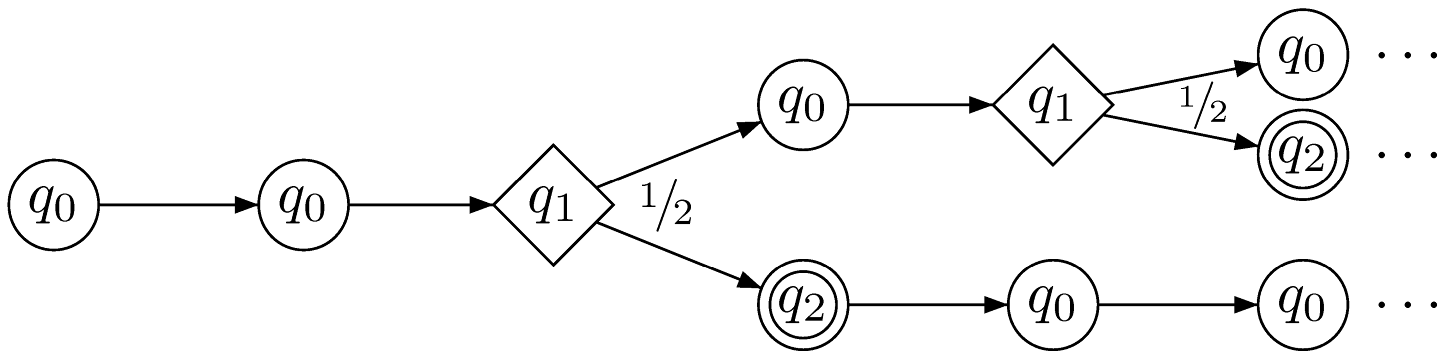
Strategy (policy) = recipe to extend the play prefix

$$\sigma : Q^* \cdot Q_\circ \to Q$$



Objective

Fix a strategy $\rightarrow$ (infinite) Markov chain



Strategy is almost-sure winning, if with probability 1:

- Büchi: visit accepting states infinitely often.

- Parity: least priority visited infinitely often is even.

Decision problem

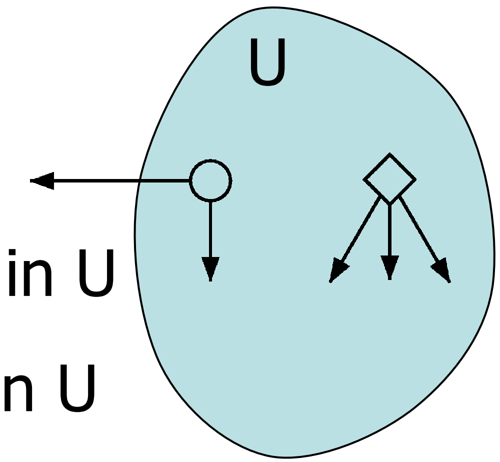
Given an MDP, decide whether there exists an almost-sure winning strategy for parity objective.

Decision problem

Given an MDP, decide whether there exists an almost-sure winning strategy for parity objective.

End-component = set of states U s.t.

- if $q \in U \cap Q_\circ$ then **some** successor of q is in U
- if $q \in U \cap Q_\diamond$ then **all** successors of q are in U
- strongly connected



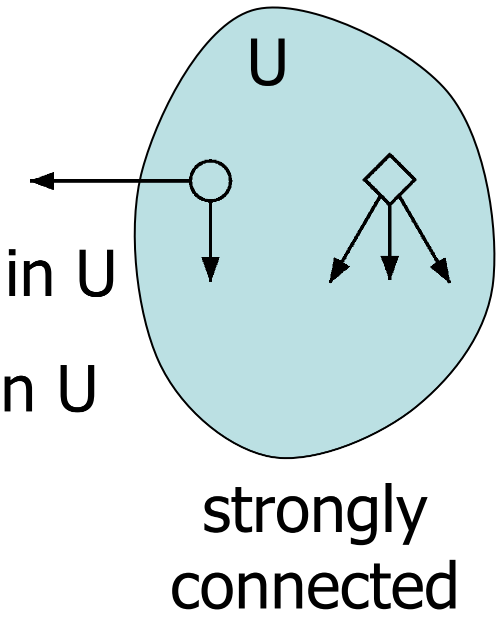
strongly connected

Decision problem

Given an MDP, decide whether there exists an almost-sure winning strategy for parity objective.

End-component = set of states U s.t.

- if $q \in U \cap Q_\circ$ then **some** successor of q is in U
- if $q \in U \cap Q_\diamond$ then **all** successors of q are in U
- strongly connected



End-component is good if least priority is even

Almost-sure reachability to good end-components in PTIME

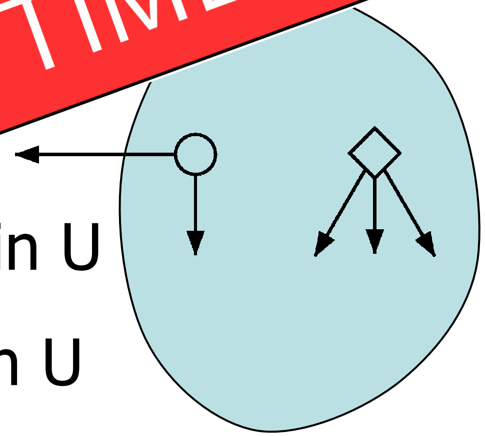
Decision problem

Given an MDP, decide whether there exists an almost-sure winning strategy for parity objective.

PTIME

End-component = set of states U s.t.

- if $q \in U \cap Q_{\circ}$ then **some** successor of q is in U
- if $q \in U \cap Q_{\diamond}$ then **all** successors of q are in U
- strongly connected



End-component is good if least priority is even

Almost-sure reachability to good end-components in PTIME

Objectives

Qualitative

Parity condition

ω-regular specifications
(reactivity, liveness,…)

$\Box\neg(g_1 \wedge g_2)$

$\Box(r \rightarrow \Diamond g)$

$\Box\Diamond r \rightarrow \Box\Diamond g$

Objectives

Qualitative

Parity condition

ω-regular specifications
(reactivity, liveness,…)

$\Box\neg(g_1 \wedge g_2)$

$\Box(r \rightarrow \Diamond g)$

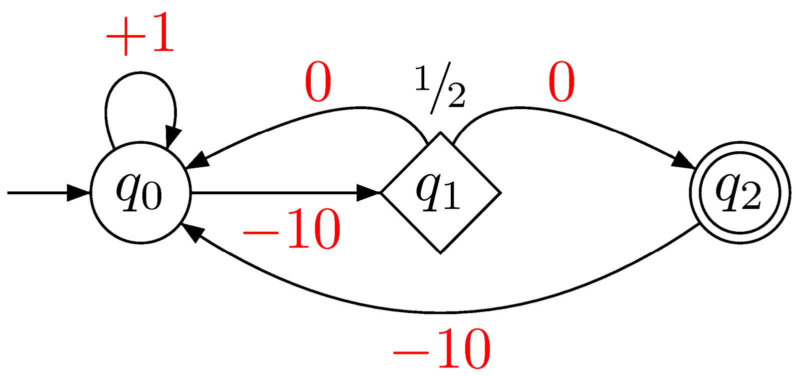
$\Box\Diamond r \rightarrow \Box\Diamond g$

Quantitative

Energy condition

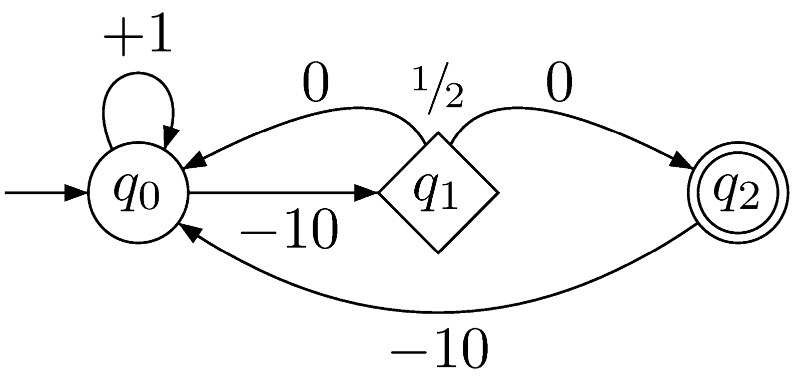
Resource-constrained
specifications

Energy objective



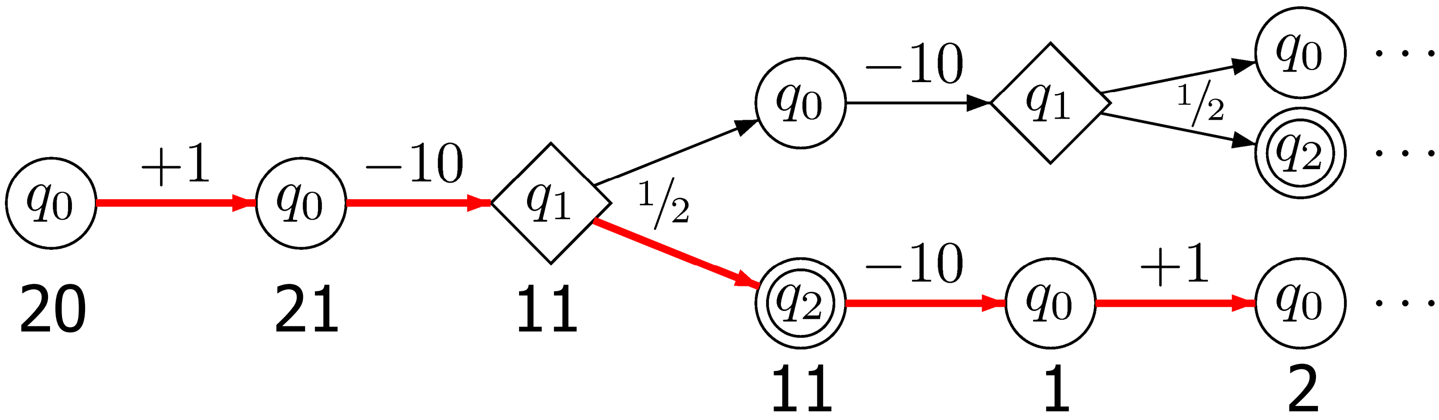
Positive and negative weights
(encoded in binary)

Energy objective

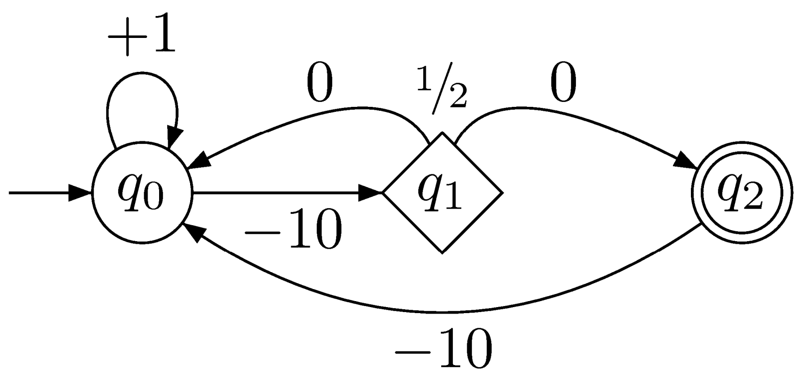


Positive and negative weights
(encoded in binary)

Energy level: 20, 21, 11, 1, 2,…
(sum of weights)



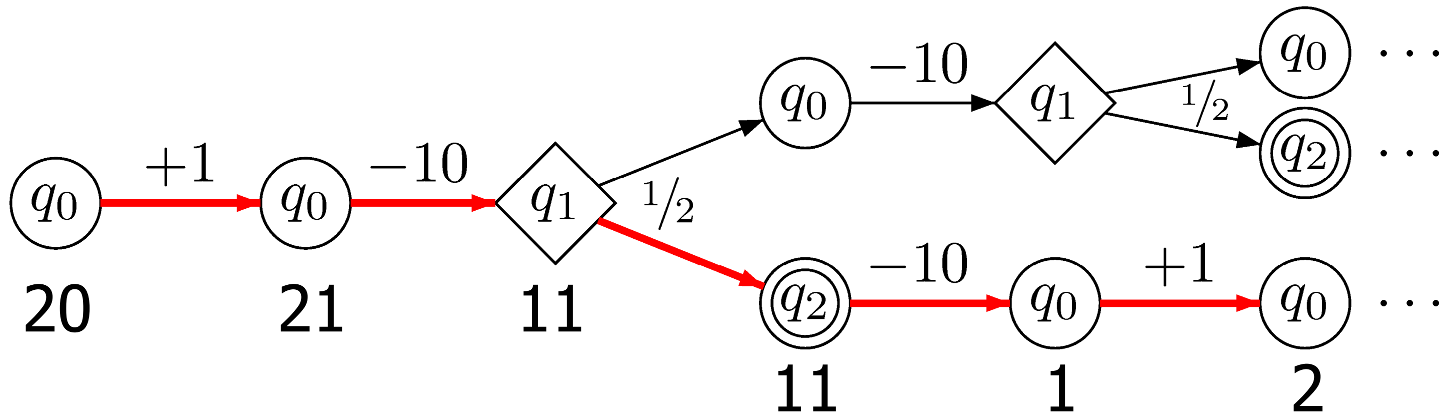
Energy objective



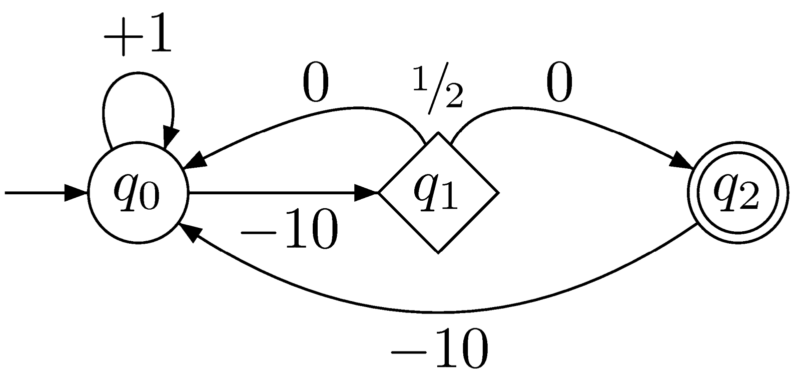
Positive and negative weights

Energy level: **20**, 21, 11, 1, 2,…
(sum of weights)

Initial credit



Energy objective

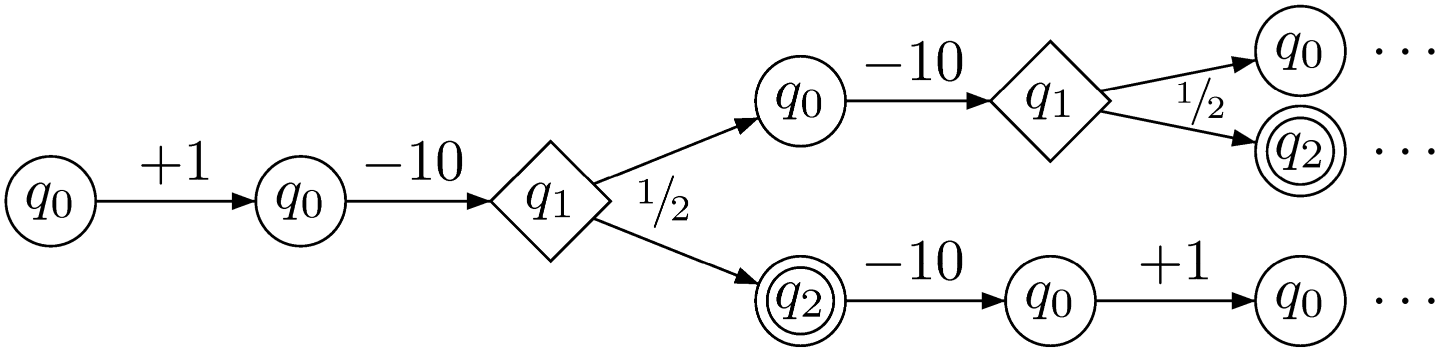


Positive and negative weights

Energy level: **20**, 21, 11, 1, 2,…
(sum of weights)

Initial credit

A play is **winning** if the energy level is always nonnegative.



"Never exhaust the resource (memory, battery, …)"

Decision problem

Given a weighted MDP, decide whether there exist an initial credit c_0 and an **almost-sure** winning strategy to maintain the energy level always nonnegative.

Decision problem

Given a weighted MDP, decide whether there exist an initial credit c_0 and an **almost-sure** winning strategy to maintain the energy level always nonnegative.

Equivalent to a two-player game:

If player 2 can force a negative energy level on a path, then the path is finite and has positive probability in MDP

Decision problem

Given a weighted MDP, decide whether there exist an initial credit c_0 and an **almost-sure** winning str
to maintain the energy level always non

NP ∩ coNP

Equivalent to a two-player gam

If player 2 can force a negative energy level on a path, then the path is finite and has positive probability in MDP

Energy Parity MDP

Objectives

Qualitative

Parity condition

PTIME

ω-regular specifications (reactivity, liveness,…)

Quantitative

Energy condition

NP ∩ coNP

Resource-constrained specifications

Mixed qualitative-quantitative

Energy parity MDP

Energy parity MDP

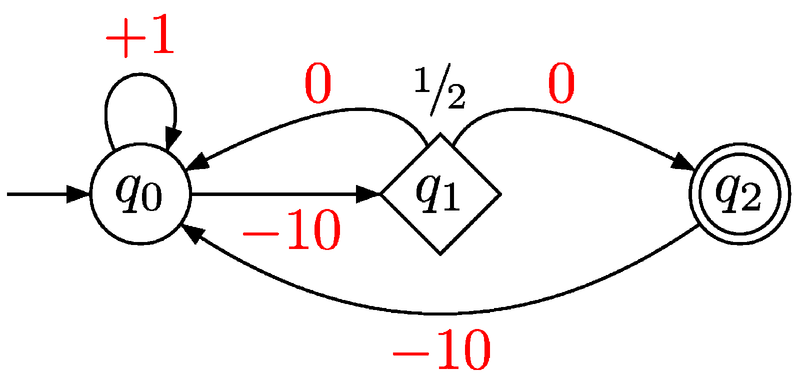
Strategy is almost-sure winning with initial credit c_0, if with probability 1:

energy condition and parity condition hold

"never exhaust the resource"

and

"always eventually do something useful"



Algorithm for Energy Büchi MDP

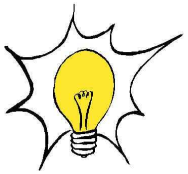
For parity, probabilistic player is my friend

For energy, probabilistic player is my opponent

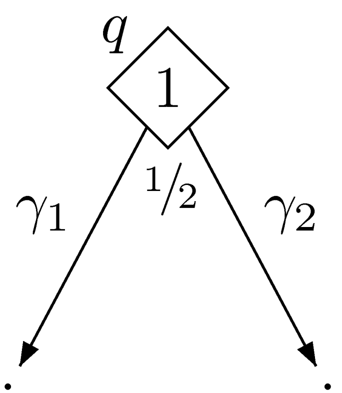
Algorithm for Energy Büchi MDP

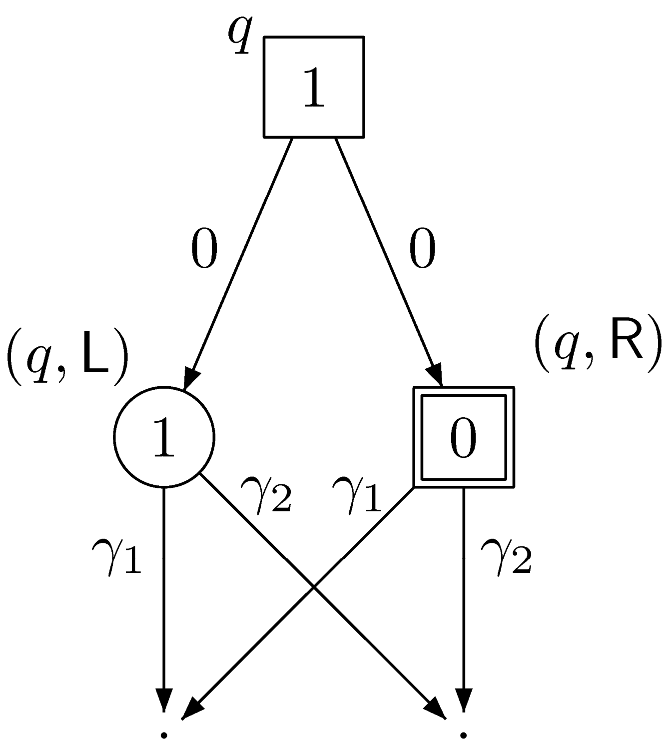
For parity, probabilistic player is my friend

For energy, probabilistic player is my opponent



Replace each probabilistic state by the gadget:





Algorithm for Energy Büchi MDP

Reduction of energy Büchi MDP to energy Büchi game

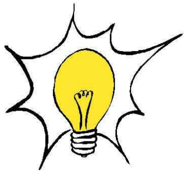
NP $\cap$ coNP

Algorithm for Energy Büchi MDP

Reduction of energy Büchi MDP to energy Büchi game

NP ∩ coNP

Reduction of energy parity MDP to energy Büchi MDP



Player 1 can guess an even priority 2i,
and win in the energy Büchi MDP where:

- Büchi states are 2i-states, and
- transitions to states with priority <2i are disallowed

NP ∩ coNP

Mean-payoff Parity MDP

Mean-payoff

Mean-payoff value of a play =
limit-average of the visited weights

$$\lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} w_i$$

Optimal mean-payoff value can be achieved with a **memoryless** strategy.

Decision problem:

Given a rational threshold ν, decide if there exists a strategy for player 1 to ensure mean-payoff value at least ν with probability 1.

Mean-payoff

Mean-payoff value of a play = limit-average of the visited weights

$$\lim_{n\to\infty} \frac{1}{n} \cdot \sum_{i=0}^{n-1} w_i$$

Optimal mean-payoff value can be achieved with a **memoryless** strategy.

Decision problem:

Given a rational threshold ν, decide if there exists a strategy for player 1 to ensure mean-payoff value at least ν with probability 1.

PTIME

Mean-payoff games

Memoryless strategy σ ensures nonnegative mean-payoff value

iff

all cycles are nonnegative in G_σ

iff

memoryless strategy σ is winning in energy game

⇒ Mean-payoff games with threshold 0 are equivalent to energy games.

Mean-payoff games

Memoryless strategy σ ensures nonnegative mean-payoff value

iff

all cycles are nonnegative in G_σ

iff

memoryless strategy σ is winning in energy game

Mean-payoff games with thresh… are equivalent to energy g…

NP ∩ coNP

Mean-payoff vs. Energy

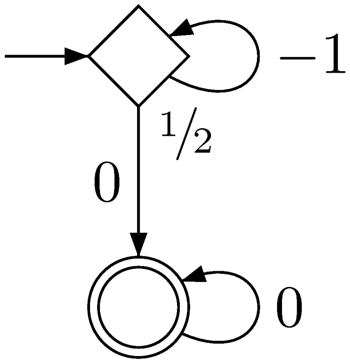
	Energy	Mean-Payoff
Games	NP ∩ coNP	NP ∩ coNP
MDP	NP ∩ coNP	PTIME

Mean-payoff parity MDPs

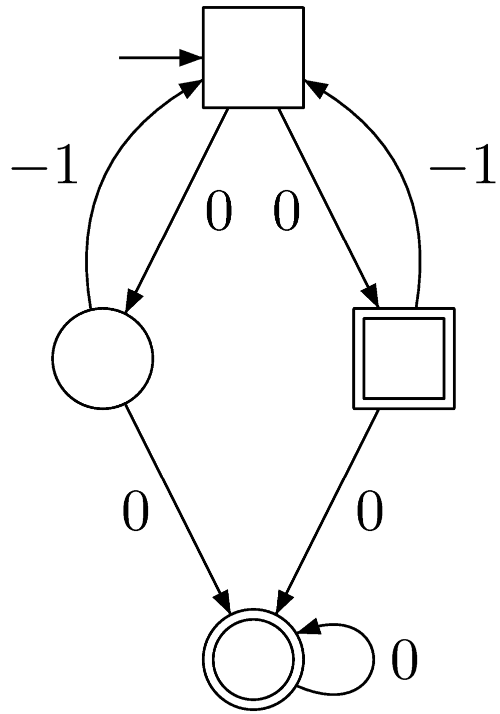
Find a strategy which ensures with probability 1:

- parity condition, and
- mean-payoff value ≥ ν

• Gadget reduction does not work:



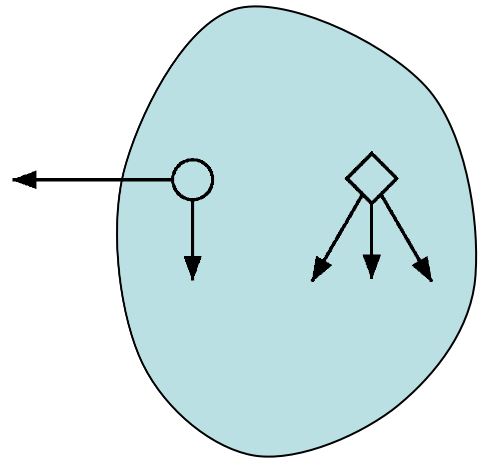
Player 1 almost-surely wins $(\nu = 0)$



Player 1 loses

Algorithm for mean-payoff parity

- End-component analysis
- almost-surely all states of end-component can be visited infinitely often
- optimal expected mean-payoff value of all states in end-component is same



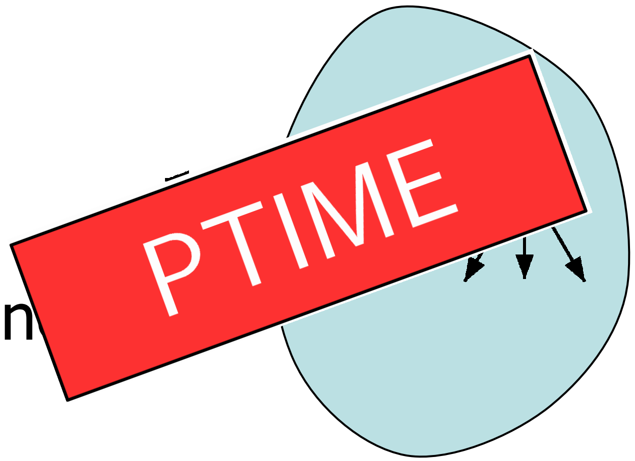
strongly connected

End-component is good if
- least priority is even
- expected mean-payoff value ≥ ν

Almost-sure reachability to good end-component in PTIME

Algorithm for mean-payoff parity

- End-component analysis
- almost-surely all states of end-compon
  can be visited infinitely often
- optimal expected mean-payoff value of all
  states in end-component is same



strongly
connected

End-component is good if
- least priority is even
- expected mean-payoff value $\geq \nu$

Almost-sure reachability to even end-component in PTIME

MDP

Qualitative

Parity MDPs — PTIME

Quantitative

Energy MDPs — NP ∩ coNP

Mean-payoff MDPs — PTIME

Mixed qualitative-quantitative

Energy parity MDPs

Mean-payoff parity MDPs

MDP

Qualitative

Parity MDPs — PTIME

Quantitative

Energy MDPs — NP ∩ coNP

Mean-payoff MDPs — PTIME

Mixed qualitative-quantitative

Energy parity MDPs — NP ∩ coNP

Mean-payoff parity MDPs — PTIME

Summary

Algorithmic complexity	Energy parity	Mean-Payoff parity
Games	NP ∩ coNP	NP ∩ coNP
MDP	NP ∩ coNP	PTIME

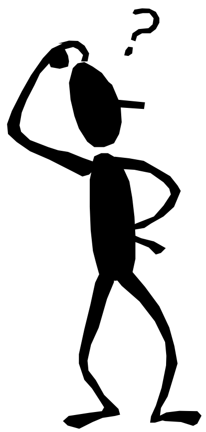
Summary

Algorithmic complexity	Energy parity	Mean-Payoff parity
Games	$NP \cap coNP$	$NP \cap coNP$
MDP	$NP \cap coNP$	PTIME

Strategy complexity	Energy parity	Mean-Payoff parity
Games	$n \cdot d \cdot W$	infinite
MDP	$2 \cdot n \cdot W$	infinite

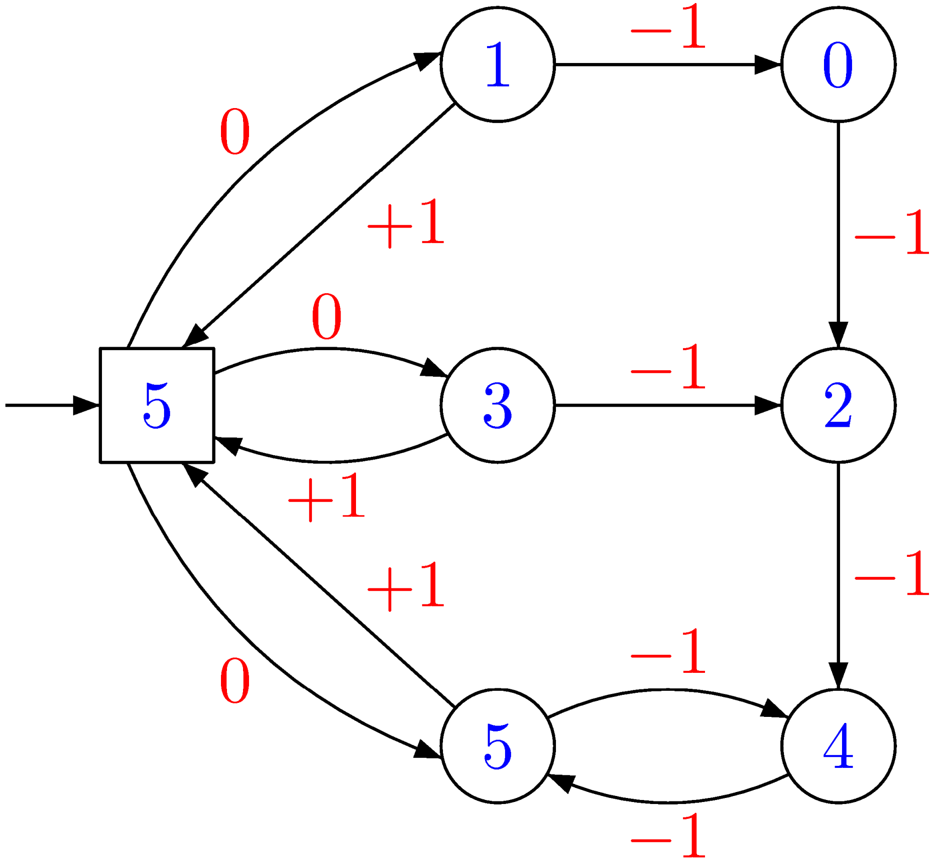
The end

Thank you !



Questions ?

The end



References

[CdAHS03] A.Chakrabarti, L. de Alfaro, T.A. Henzinger, and M. Stoelinga. **Resource interfaces**, Proc. of EMSOFT: Embedded Software, LNCS 2855, Springer, pp.117-133, 2003

[EM79] A. Ehrenfeucht, and J. Mycielski, **Positional Strategies for Mean-Payoff Games**, International Journal of Game Theory, vol. 8, pp. 109-113, 1979

[BFL+08] P. Bouyer, U. Fahrenberg, K.G. Larsen, N. Markey, and J. Srba, **Infinite Runs in Weighted Timed Automata with Energy Constraints**, Proc. of FORMATS: Formal Modeling and Analysis of Timed Systems, LNCS 5215, Springer, pp. 33-47, 2008

[CHJ05] K. Chatterjee, T.A. Henzinger, M. Jurdzinski. **Mean-payoff Parity Games**, Proc. of LICS: Logic in Computer Science, IEEE, pp. 178-187, 2005.

Energy parity

Algorithm 1: SolveEnergyParityGame

Input : An energy parity game $\langle G, p, w \rangle$ with state space Q.

Output: The set of winning states in $\langle G, p, w \rangle$ for player 1.

begin
- **if** $Q = \varnothing$ **then return** $\varnothing$;
- Let k^* be the minimal priority in G. Assume w.l.o.g. that $k^* \in \{0, 1\}$;
- Let G_0 be the game G ;
- $i \leftarrow 0$;
- **if** $k^* = 0$ **then**
  - $A_0 \leftarrow Q$ `/* over-approximation of Player-1 winning states */` ;
  - **repeat**
    - $A'_i \leftarrow$ SolveEnergyGame(G_i, w') (where w' is defined in Lemma **??**) ;
    - $X_i \leftarrow Attr_1(A'_i \cap p^{-1}(0))$;
    - Let G'_i be the subgraph of G_i induced by $A'_i \setminus X_i$;
    - $Z_i \leftarrow (A'_i \setminus X_i) \setminus$ SolveEnergyParityGame(G'_i, p, w) ;
    - $A_{i+1} \leftarrow A'_i \setminus Attr_2(Z_i)$;
    - Let G_{i+1} be the subgraph of G_i induced by A_{i+1} ;
    - $i \leftarrow i + 1$;
  - **until** $A_i = A_{i-1}$;
  - **return** A_i;
- **if** $k^* = 1$ **then**
  - $B_0 \leftarrow Q$ `/* over-approximation of Player-2 winning states */` ;
  - **repeat**
    - $Y_i \leftarrow Attr_2(B_i \cap p^{-1}(1))$;
    - Let G_{i+1} be the subgraph of G_i induced by $B_i \setminus Y_i$;
    - $B_{i+1} \leftarrow B_i \setminus Attr_1($SolveEnergyParityGame$(G_{i+1}, p, w))$;
    - $i \leftarrow i + 1$;
  - **until** $B_i = B_{i-1}$;
  - **return** $Q \setminus B_i$;

end

Mean-payoff parity

Algorithm 1 ComputeLeastValueClass

Input: a mean-payoff parity game $\mathcal{MP} = (\mathcal{G}, p, r)$ such that $p^{-1}(0) \neq \emptyset$ and the game is parity winning for player 1.
Output: a nonempty 1-closed subset of $\mathcal{LV}$, and $MP_1(v)$ for all $v \in \mathcal{LV}$.
1. $F = Attr_1(p^{-1}(0), \mathcal{G})$.
2. $H = V \setminus F$ and $\mathcal{H} = \mathcal{G} \upharpoonright H$.
3. **MeanPayoffParitySolve**($\mathcal{H}$) (Algorithm 3).
4. Construct the mean-payoff game $\widetilde{\mathcal{G}}$ as described in Subsection 3.1 and **Solve**
5. Let $\mathcal{LV}_{\widetilde{\mathcal{G}}}$ be the least value class in $\widetilde{\mathcal{G}}$ and $\widetilde{l}$ be the least value.
6. $\mathcal{LV} = \mathcal{LV}_{\widetilde{\mathcal{G}}} \cap V$, and $MP_1(v) = \widetilde{l}$ for all $v \in \mathcal{LV}$.
7. **return** $(\mathcal{LV}, \widetilde{l})$.

Subroutine *SetValues*(J_i, j_i)

1. $g = \max\{\, Val(w) : w \in W_0$ and $\exists v \in J_i \cap V_1.\ (v, w) \in E\}$.
2.1 **if** $g > j_i$ **then**
  2.2 $T_1 = \{v \in J_i \cap V_1 : \exists w \in W_0.\ Val(w) = g$ and $(v, w) \in E\}$; and $W_0 = W_0 \cup UnivReach(T_1)$.
  2.3 For every vertex $v \in UnivReach(T_1)$, set $Val(v) = g$.
  2.4 **goto** Step 6.3. of **MeanPayoffParitySolve**.
3. $l = \min\{\, Val(w) : w \in W_0$ and $\exists v \in J_i \cap V_2.\ (v, w) \in E\}$.
4.1 **if** $l < j_i$ **then**
  4.2 $T_2 = \{v \in J_i \cap V_2 : \exists w \in W_0.\ Val(w) = l$ and $(v, w) \in E\}$; and $W_0 = W_0 \cup UnivReach(T_2)$.
  4.3 For every vertex $v \in UnivReach(T_2)$, set $Val(v) = l$.
  4.4 **goto** Step 6.3. of **MeanPayoffParitySolve**.

Algorithm 2 ComputeGreatestValueClass

Input: a mean-payoff parity game $\mathcal{MP} = (\mathcal{G}, p, r)$ such that $p^{-1}(0) = \emptyset$ and $p^{-1}(1) \neq \emptyset$, and the game is parity winning for player 1.
Output: a nonempty
1. $F = Attr_2(p^{-1}(1$
2. $H = V \setminus F$ and $\mathcal{H}$
3. **MeanPayoffParit**
4. Let $\mathcal{GV}_{\mathcal{H}}$ be the gr
5. $\mathcal{GV} = \mathcal{GV}_{\mathcal{H}}$, and M
6. **return** $(\mathcal{GV}, \hat{g})$.

Algorithm 3 MeanPayoffParitySolve

Input: a mean-payoff party game $\mathcal{MP}$. **Output:** the value function MP_1.
1. Compute W_1 and W_2 by any algorithm for solving parity games.
2. For every vertex $v \in W_2$, set $Val(v) = -\infty$. 3. $W_0 = \emptyset$. 4. $\mathcal{G}_0 = \mathcal{G} \upharpoonright W_1$ and $V^0 = W_1$. 5. $i = 0$.
6. **repeat**
  6.1. **while** $(p^{-1}(0) \cup p^{-1}(1)) \cap V_i = \emptyset$ **do** set $p = p - 2$. **end while**
  6.2. Let (W_1^i, W_2^i) be the partition of the parity winning sets in $\mathcal{G}_i$.
  6.2.a. **if** $W_2^i \neq \emptyset$ **then**
    6.2.a.1. $g = \max\{\, Val(w) : w \in W_0$ and $\exists v \in W_2^i \cap V_1.\ (v, w) \in E\}$.
    6.2.a.2. $T_1 = \{v \in L_i \cap V_1 : \exists w \in W_0.\ Val(w) = g$ and $(v, w) \in E\}$.
    6.2.a.3. $W_0 = W_0 \cup UnivReach(T_1)$.
    6.2.a.4. For every vertex $v \in UnivReach(T_1)$, set $Val(v) = g$.
    goto Step. 6.3.
  6.2.b. **else**
    6.2.b.1. **if** $p^{-1}(0) \cap V_i \neq \emptyset$ **then** let $(L_i, l_i) =$ **ComputeLeastValueClass**$(\mathcal{G}_i)$.
      6.2.b.1.a. Subroutine *SetValues*(L_i, l_i)
      6.2.b.1.b. $W_0 = W_0 \cup L_i$, and for every vertex $v \in L_i$, set $Val(v) = l_i$.
    6.2.b.2. **else** $(G_i, g_i) =$ **ComputeGreatestValueClass**$(\mathcal{G}_i)$.
      6.2.b.2.a. Subroutine *SetValues*(G_i, g_i)
      6.2.b.2.b. $W_0 = W_0 \cup G_i$, and for every vertex $v \in G_i$, set $Val(v) = g_i$.
  6.3. $V^{i+1} = V^i \setminus W_0$ and $\mathcal{G}_i = \mathcal{G} \upharpoonright V^i$.
  6.4. $i = i + 1$.
until $V_i = \emptyset$ **(end repeat)**
7. $MP_1 = Val$.

Complexity

	Strategy		Algorithmic complexity
	Player 1	Player 2	
Energy	memoryless	memoryless	NP ∩ coNP
Parity	memoryless	memoryless	NP ∩ coNP
Energy parity	exponential	memoryless	NP ∩ coNP
Mean-payoff	memoryless	memoryless	NP ∩ coNP
Mean-payoff parity	infinite	memoryless	NP ∩ coNP