A faster deterministic exponential time algorithm for Energy Games and Mean Payoff

Dani Dorfman Haim Kaplan Uri Zwick

Tel Aviv University

Abstract

We present an improved exponential time algorithm for Energy games, and hence also for Mean Payoff Games. The running time of the new algorithm is $O(\min(mnW, mn2^{\frac{n}{2}} \log W))$, where n is the number of vertices, m is the number of edges, and when the edge weights are assumed to be integers of absolute value at most W. The new algorithm is also a pseudopolynomial time algorithm, matching the running time of the fastest known pseudopolynomial time algorithm of Brim et al. on which it is based. The new algorithm is obtained by introducing a technique of forecasting repetitive actions performed by the algorithm of Brim et al., along with the use of a *scaling* technique.

State of the Art

Algorithm	*Time*	*Det/Ran*
This paper	$O(\min(nmW, nm2^{\frac{n}{2}} \log W)$	Deterministic
Brim et al.	$O(nmW)$	Deterministic
Lifshits & Pavlov	$O(nm2^n)$	Deterministic
Bjorklund & Vorobyov	$2^{O(\sqrt{n \log n})}$	Randomized

Preliminaries

Game Graphs a *game graph* is a tuple $\Gamma = (V, E, w, < V_0, V_1 >)$ where $< V_0, V_1 >$ is a partition of V into the set of player-0 vertices and the set of player-1 vertices, respectively. We call a mapping ϕ_i: $V_i \rightarrow V$ *positional strategy* of player-i if for all $v \in V_i$ it holds that $(v, \phi_i(v)) \in E$. Given *positional strategies* ϕ_0, ϕ_1 of player-0 and player-1 and an initial vertex v_0, $play(v_0, \phi_0, \phi_1) = v0, v1, \ldots, vi, \ldots$ is the infinite walk resulted by 0 and 1 starting at v_0.

Energy-Games an *energy-game* is an infinite game on a game graph Γ, where the goal of player-0, when starting at some $v_0 \in V$, is to device a *positional strategy* ϕ_0 such that for all *positional strategies* ϕ_1 of player-1 the following holds for some initial credit $c \in \mathbb{N}$:

$$\forall j \geq 0, \qquad c + \sum_{i=0}^{j} w(v_i, v_{i+1}) \geq 0$$

Goal find minimal initial credits $c: V \rightarrow \mathbb{N} \cup \{\infty\}$ ($c(v) = \infty$ iff v wins for player-1).

Energy Progress Measures

A function $f: V \rightarrow \mathbb{N} \cup \{\infty\}$ is an $energy\ progress\ measure$ iff for every $v \in V$ the following holds:

- If $v \in V_0$ then $f(v) + w(v,u) \geq f(u)$ for $some\ u \in V$.
- If $v \in V_1$ then $f(v) + w(v,u) \geq f(u)$ for $all\ u \in V$.

A vertex is $Valid$ with respect to a function $g: V \rightarrow \mathbb{N} \cup \{\infty\}$ if the above constraint holds.

Lemma: Let $f: V \rightarrow \mathbb{N} \cup \{\infty\}$ be an energy progress measure, then for every $v \in V$, if $f(v) < \infty$ then v is winning for player-0.

Lemma: Let $g(v) = \min\{f(v) | f$ is an energy progress measure$\}$, then g is an energy progress measure and g is the minimal initial credit function.

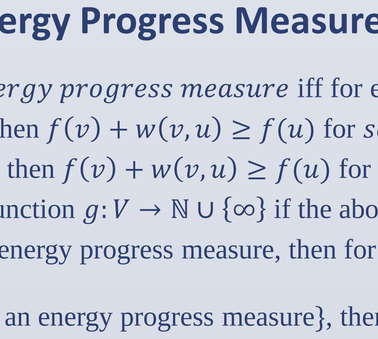
The algorithm of Brim et al.

Algorithm 1: $compute\text{–}energy\,(V, E, w, \langle V_0, V_1 \rangle)$

```
1 foreach v ∈ V do
2 |   f(v) ← 0
3 while ∃ v Invalid do
4 |   Increase f(v) s.t v is Valid
5 if f(v) > T then f(v) ← ∞
6 return f
```

It is enough to set the threshold $T = nW$. Thus, an $O(mnW)$ time algorithm.

Worst Case Example:



The Improved Algorithm

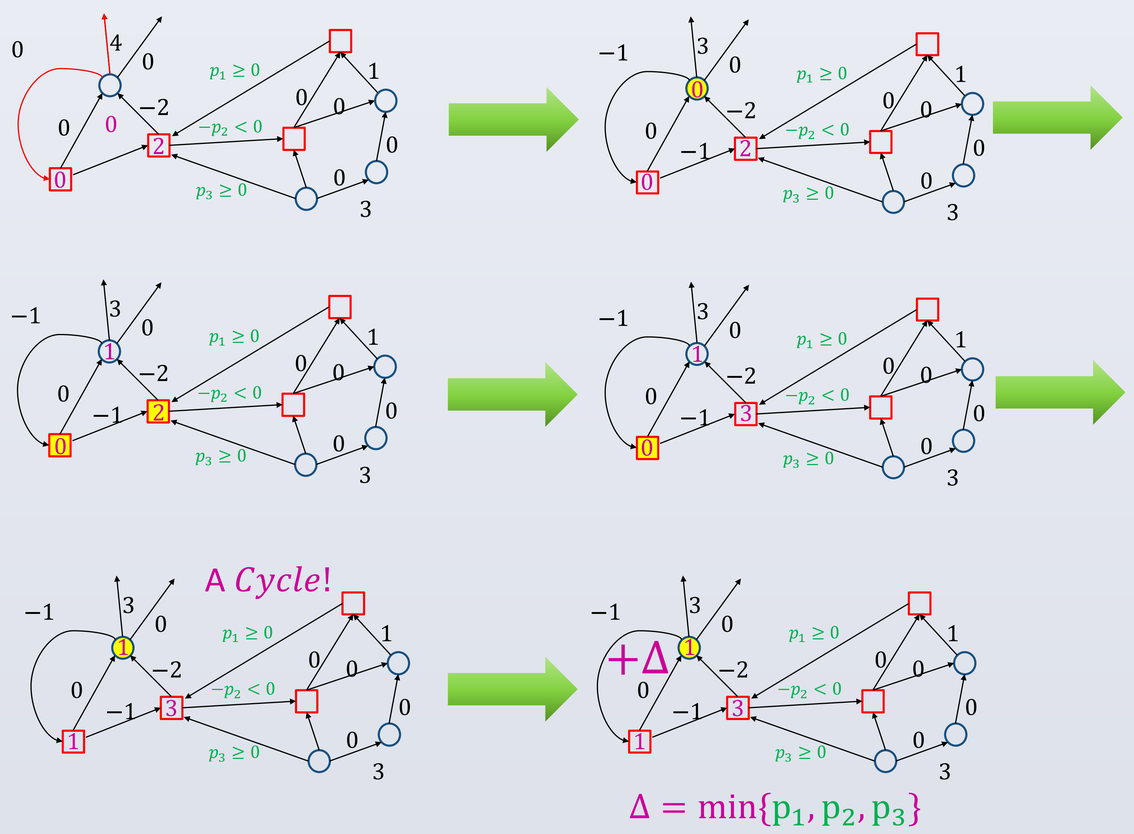
As in the algorithm of Brim et al., we maintain a function $f: V \rightarrow \mathbb{N}$ starting with the 0 function. The algorithm starts with a standard scaling approach (see pseudo-code below). When rescaling the edge weights we scan the vertices one by one, fix their edge weights and update f to be an $energy\ progress\ measure$ of the new game ($update - energy()$).

Algorithm 2: $compute\text{–}energy\,(V, E, w, \langle V_0, V_1 \rangle)$

```
1  if w ≥ 0 then
2  |   return f ≡ 0
3  w' ← ⌈w/2⌉
4  f ← compute–energy (V, E, w', ⟨V0, V1⟩)
5  f, w' ← 2f, 2w'
6  foreach v ∈ V do
7  |   foreach e ∈ out(v) do
8  |   |   if w'(e) > w(e) then w'(e) ← w'(e) − 1
9  |   update–energy (V, E, w', ⟨V0, V1⟩ , f, v)
10 return f
```

Algorithm 3: $update\text{–}energy\,(V, E, w, \langle V_0, V_1 \rangle, f, v)$

```
1  if v ∈ V0 and ∀(v,u) ∈ E : wf(v,u) < 0 then L ← {v}
2  if v ∈ V1 and ∃(v,u) ∈ E : wf(v,u) < 0 then L ← {v}
3  foreach u ∈ V0 do
4  |   count[u] ← |{u' | (u,u') ∈ E, wf(u,u') ≥ 0}|
5  while L = {v} and f(v) ≤ 4 · |V| · W do
6  |   B ← {v}
7  |   update(v, L, B)
8  |   while L \ {v} ≠ ∅ do
9  |   |   pick u ∈ L \ {v}
10 |   |   update(u, L, B)
11 |   Δ ← delta(B)
12 |   foreach u ∈ B do f(u) ← f(u) + Δ
13 foreach u ∈ V such that f(u) ≥ |V| · W do f(u) ← ∞
```



$update - energy()$ performs as follows. Let v be the vertex that his edge weights were decremented. If v is still $Valid$ then we are done. Otherwise, increment $f(v)$ by 1. This makes v $valid$ but vertices u with edges (u, v) may become invalid. We keep on incrementing $f(u)$ of all $invalid$ vertices $u \neq v$ until either all vertices are $valid$ (and we are done - f is an $energy$ progress measure) or v is the only $invalid$ vertex. Assume that v became $invalid$ and let B be the set of vertices that we fixed. When we fix v again it could be that the same set of vertices B will become invalid again or it could be that a different set of vertices $B' \neq B$ will become invalid . We compute Δ, the number of times that the set B repeats itself and increase $f(u)$ of all $u \in B$ by Δ.

The above method of forecasting cycles and updating f accordingly is the key component of our algorithm.

ACKNOWLEDGEMENTS

I would like to thank the Deutch Grant for supporting this paper.

References

[BCD+11] Lubos Brim, Jakub Chaloupka, Laurent Doyen, Raaella Gentilini, and Jean-Francois Raskin. Faster algorithms for mean-payo games. Formal methods in system design, 38(2):97{118, 2011.

[BFL+08] Patricia Bouyer, Uli Fahrenberg, Kim G Larsen, Nicolas Markey, and Jir Srba. Innite runs in weighted timed automata with energy constraints. In International Conference on Formal Modeling and Analysis of Timed Systems, pages 33{47. Springer, 2008.

[BV07] Henrik Bj orklund and Sergei Vorobyov. A combinatorial strongly subexponential strategy improvement algorithm for mean payo games. Discrete Applied Mathematics, 155(2):210{229, 2007.

[LP07] Yuri M Lifshits and Dmitri S Pavlov. Potential theory for mean payo games. Journal of Mathematical Sciences, 145(3):4967{4974, 2007.

[ZP96] Uri Zwick and Mike Paterson. The complexity of mean payo games on graphs. Theoretical Computer Science, 158(1-2):343{359, 1996.