The Complexity of Mean Payoff Games *

Uri Zwick[1] and Michael S. Paterson[2]

[1] Dept. of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel
[2] Dept. of Computer Science, Univ. of Warwick, Coventry CV4 7AL, UK

Abstract. We study the complexity of finding the values and optimal strategies of *mean payoff games*, a family of perfect information games introduced by Ehrenfeucht and Mycielski. We describe a pseudo-polynomial time algorithm for the solution of such games, the decision problem for which is in NP ∩ co-NP. Finally, we describe a polynomial reduction from mean payoff games to the *simple stochastic games* studied by Condon. These games are also known to be in NP ∩ co-NP, but no polynomial or pseudo-polynomial time algorithm is known for them.

1 Introduction

Let $G = (V, E)$ be a finite directed graph in which each vertex has at least one edge going out of it. Let $w : E \to \{-W, \ldots, 0, \ldots, W\}$ be a function that assigns an integral weight to each edge of G. Ehrenfeucht and Mycielski [EM79] studied the following infinite two-person game played on such a graph. The game starts at a vertex $a_0 \in V$. The first player chooses an edge $e_1 = (a_0, a_1) \in E$. The second player then chooses an edge $e_2 = (a_1, a_2) \in E$, and so on indefinitely. The first player wants to maximise $\liminf_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} w(e_i)$. The second player wants to minimise $\limsup_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} w(e_i)$. Ehrenfeucht and Mycielski show that each such game has a value ν such that the first player has a strategy that ensures that $\liminf_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} w(e_i) \geq \nu$, while the second player has a strategy that ensures that $\limsup_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} w(e_i) \leq \nu$. Furthermore, they show that both players can achieve this value using a *positional strategy*, i.e., a strategy in which the next move depends only on the vertex from which the player is to move.

Without loss of generality, we may assume that the graph $G = (V, E)$ on which such a game is played is bipartite, with V_1 and V_2 being the partition of the vertices into the two 'sides' and with $E = E_1 \cup E_2$ such that $E_1 \subseteq V_1 \times V_2$ and $E_2 \subseteq V_2 \times V_1$. If the original graph is not bipartite, we simply duplicate the set of vertices.

To obtain their results for the infinite game, Ehrenfeucht and Mycielski [EM79] also consider the following finite version of the game. Again the game starts at a specific vertex of the graph, which is assumed to be bipartite. The

* Supported in part by the ESPRIT Basic Research Action Programme of the EC under contract No. 7141 (project ALCOM II). E-mail addresses of authors: **zwick@math.tau.ac.il** and **Mike.Paterson@dcs.warwick.ac.uk**.

players alternate in choosing successive edges that form a path, but the game ends as soon as a cycle is formed. The outcome of the game is then the mean weight of the edges on this cycle. The first player wants to maximise and the second player to minimise this outcome. This game is a finite perfect-information two-person game and so, by definition, has a value. Ehrenfeucht and Mycielski [EM79] show that the value ν of this finite game is also the value of the infinite game described above. Furthermore, they show, surprisingly perhaps, that both players have positional optimal strategies for the finite game. The positional optimal strategies of the finite game are also positional optimal strategies for the infinite game.

Ehrenfeucht and Mycielski [EM79] give no efficient algorithm for finding optimal strategies for the finite and infinite games. We complement their work by exhibiting an $O(|V|^3 \cdot |E| \cdot W)$ time algorithm for finding the values of the games played on a graph $G = (V, E)$. The algorithm finds the values of all the vertices of the graph. Games starting at different vertices may have different values, of course. We also give an $O(|V|^4 \cdot |E| \cdot \log(|E|/|V|) \cdot W)$ time algorithm for finding positional optimal strategies for both players. Our algorithm is polynomial in the size of the graph but only pseudo-polynomial in the weights. Our algorithm is polynomial if the weights are presented in unary notation. In particular, our algorithms work in polynomial time if the weights are taken from, say, $\{-1, 0, +1\}$. This is already a non-trivial case.

We also consider situations in which one player knows in advance the positional strategy to be used by the other player. Using a result of Karp [Kar78] we show that an optimal counter-strategy can be found in (strongly) polynomial time. This immediately implies that the decision problem associated with the game is in NP $\cap$ co-NP.

The decision problem corresponding to mean payoff games (MPG's) is thus in NP $\cap$ co-NP as well as in $\tilde{\text{P}}$ (pseudo-polynomial time), but is not yet known to be in P. This gives the MPG problem a rare status shared only by a few number-theoretic problems, such as *primality* [Pra75].

Mean payoff games have been considered independently by Gurvich, Karzanov and Khachiyan [GKK88]. They were not aware of the results of Ehrenfeucht and Mycielski and gave an alternative proof of the fact that both players in mean payoff games, or *cyclic games* as they call them, have positional optimal strategies. Gurvich *et al.* give an algorithm for finding such optimal strategies, but the worst-case complexity of their algorithm is exponential. Further generalisations and variants of mean payoff games have also been considered by Karzanov and Lebedev [KL93], who also point out that the decision problem corresponding to mean payoff games is in NP $\cap$ co-NP.

Condon [Con92] has recently studied the complexity of *simple stochastic games* (SSG's) introduced originally by Shapley [Sha53]. Condon shows that the decision problem corresponding to SSG's is also in NP $\cap$ co-NP. While MPG's are deterministic, SSG's are games of chance. We describe a simple reduction from MPG's to SSG's in two steps. We first describe a reduction from MPG's to *discounted payoff games* (DPG's), and then a reduction from DPG's to SSG's.

The reduction from MPG's to SSG's shows that SSG's are at least as hard as MPG's. We believe that the MPG problem is strictly easier then the SSG problem. As attempts to obtain polynomial time algorithms for SSG's have not yet borne fruit, it may be interesting to focus attention on the possibly easier problem of obtaining a polynomial time algorithm for MPG's.

2 Finding the values of a game

Let $G = (V, E)$ be a graph and let $w : E \to \{-W, \ldots, 0, \ldots, W\}$ be a weight function on its edges. Let $|V| = n$. We assume that the graph is bipartite with V_1 being the set of vertices from which player I is to play, and V_2 being the set of edges from which player II is to play.

Our first goal is to find, for each vertex $a \in V$, the value $\nu(a)$ of the finite and infinite games that start at a. Recall that the values of the finite and infinite games are equal. If $a \in V_1$ then player I (the maximiser) is to play first and if $a \in V_2$ then the second player (the minimiser) is to play first.

To reach this goal we consider a third version of the game. This time the two players play the game for exactly k steps constructing a path of length k, and the weight of this path is the outcome of the game. The length of the game is known is advance to both players. We let $\nu_k(a)$ be the value of this game started at $a \in V$, where player I or II plays first according to whether $a \in V_1$ or $a \in V_2$.

Theorem 1. *The values $\nu_k(a)$, for all $a \in V$, can be computed in $O(k \cdot |E|)$ time.*

Proof. The result follows easily from the following recursive relation

$$\nu_k(a) = \begin{cases} \max_{(a,b)\in E}\{w(a,b) + \nu_{k-1}(b)\} & \text{if } a \in V_1 , \\ \min_{(a,b)\in E}\{w(a,b) + \nu_{k-1}(b)\} & \text{if } a \in V_2 , \end{cases}$$

along with the initial condition, $\nu_0(a) = 0$ for every $a \in V$. □

It seems intuitively clear that $\lim_{k\to\infty} \nu_k(a)/k = \nu(a)$, where $\nu(a)$ is the value of the infinite game that starts at a. The next theorem shows that this is indeed the case. In the proof of this theorem we rely on the result, proved by Ehrenfeucht and Mycielski, that both players have positional optimal strategies. A *positional strategy for player I* is just a mapping $\pi_1 : V_1 \to V_2$ such that $(a_1, \pi_1(a_1)) \in E_1$ for every $a_1 \in V_1$. Similarly, a *positional strategy for player II* is a mapping $\pi_2 : V_2 \to V_1$ such that $(a_2, \pi_2(a_2)) \in E_2$ for every $a_2 \in V_2$.

Theorem 2. *For every $a \in V$ we have: $k\cdot\nu(a) - 2nW \le \nu_k(a) \le k\cdot\nu(a) + 2nW$.*

Proof. Let $\pi_1 : V_1 \to V_2$ be a positional optimal strategy for player I in the finite game starting at a. We show that if player I plays using the strategy π_1 then the outcome of a k-step game is at least $(k - n)\cdot\nu(a) - nW$. Consider a game in which player I plays according to π_1. Push (copies of) the edges played by the players onto a stack. Whenever a cycle is formed, it follows from the fact that π_1 is an optimal strategy for player I in the finite game, that the mean weight of the cycle formed is at least $\nu(a)$. The edges that participate in that cycle lie consecutively at the top of the stack. They are all removed and the process continues. Note that at each stage the stack contains at most n edges and the weight of each of them is at least $-W$. Player I can therefore ensure that the total weight of the edges encountered in a k-step game starting from a is at least $(k-n)\cdot\nu(a) - nW$. This is at least $k\cdot\nu(a) - 2nW$ as $\nu(a) \le W$.

Similarly, if player II plays according to a positional optimal strategy $\pi_2 : V_2 \to V_1$ of the finite game that starts at a, she can make sure that the mean weight of each cycle closed is at most $\nu(a)$. At most n edges are left on the stack and the weight of each of them is at most W. She can therefore ensure that the total weight of the edges encountered in a k-step game starting at a is at most $(k-n)\cdot\nu(a) + nW \le k\cdot\nu(a) + 2nW$. □

We can now describe the algorithm for computing the exact values of the finite and infinite games.

Theorem 3. *Let $G = (V, E)$ be a directed graph and let $w : E \to \{-W, \ldots, 0, \ldots, W\}$ be a weight function on its edges. The value $\nu(a)$, for every $a \in V$, corresponding to the infinite and finite games that start at all the vertices of V can be computed in $O(|V|^3\cdot|E|\cdot W)$ time.*

Proof. Compute the values $\nu_k(a)$, for every $a \in V$, for $k = 2n^3W$. This can be done, according to Theorem 1, in $O(|V|^3\cdot|E|\cdot W)$ time. For each vertex $a \in V$, compute the estimate $\nu'(a) = \nu_k(a)/k$. By Theorem 2, we get that

$$\nu'(a) - \frac{1}{n(n-2)} < \nu'(a) - \frac{2nW}{k} \le \nu(a) \le \nu'(a) + \frac{2nW}{k} < \nu'(a) + \frac{1}{n(n-2)} \,.$$

The value $\nu(a)$ is a rational number, with an even denominator whose size is at most n. The minimum distance between two possible values of $\nu(a)$ is at least $2/(n(n-2))$. The exact value of $\nu(a)$ is therefore the unique rational number with an even denominator of size at most n that lies in the interval $(\nu'(a) - \frac{1}{n(n-2)}, \nu'(a) + \frac{1}{n(n-2)})$. This number is easily found. □

Slightly less accuracy is needed if we just want to know whether the value of each position is negative, zero or positive. This decision problem can therefore be decided more efficiently.

Theorem 4. *Let G and w be as in Theorem 3, and let T be an integer threshold. The decision whether $\nu(a) < T$, $\nu(a) = T$, or $\nu(a) > T$, for every $a \in V$, can be made in $O(|V|^2\cdot|E|\cdot W)$ time.*

Proof. The distance between T and the closest rational number with an even denominator of size at most n is at least $1/n$. It is therefore enough to compute the values $\nu_k(a)$ for $k = 4n^2W$, and this takes only $O(|V|^2\cdot|E|\cdot W)$ time. □

3 Finding the optimal strategies

Given an algorithm for finding the value of any vertex of a graph, positional optimal strategies can be found using a simple method.

Theorem 5. *Let $G = (V, E)$ be a directed graph and let $w : E \rightarrow \{-W, \ldots, 0, \ldots, W\}$ be a weight function on its edges. Positional optimal strategies for both players for games played on G can be found in $O(|V|^4 \cdot |E| \cdot \log(|E|/|V|) \cdot W)$ time.*

Proof. Start by computing the values $\nu(a)$ for every $a \in V$. If all the vertices $a \in V_1$ have outdegree one, then player I has a unique strategy and this strategy is positional and optimal. Otherwise, consider any vertex $a \in V_1$ with outdegree $d > 1$. Remove any $\lceil d/2 \rceil$ of the edges leaving a, and recompute the value of a, $\nu'(a)$ say, for the resulting graph. If $\nu'(a) = \nu(a)$ then there is a positional optimal strategy for the player I which does not use any of the removed edges; if $\nu'(a) \neq \nu(a)$ then there is a positional optimal strategy for this player using one of the removed edges. Whichever is the case, we can now restrict attention to a subgraph G' with at least $\lfloor d/2 \rfloor$ fewer edges. Let $d(a)$ be the initial outdegree of a vertex $a \in V$. After $O(\sum_{a \in V_1} \log d(a))$ such experiments we are left with a positional optimal strategy for player I. A positional optimal strategy for player II is found in a similar way. As $\sum_{a \in V} \log d(a) \leq |V| \cdot \log(|E|/|V|)$, we get that the complexity of this algorithm is $O(|V|^4 \cdot |E| \cdot \log(|E|/|V|) \cdot W)$, as required. □

An interesting open problem is whether finding positional optimal strategies is harder than just computing the values of a game. The algorithm we describe calls the full value-finding algorithm repeatedly, but uses only the value at a single vertex and ignores any information about the optimal moves of the players in the truncated games. Unfortunately, optimal moves in the truncated games may not conform to positional strategies. We think however that it should be possible to use the additional information gathered and improve our algorithm.

4 Playing against a known positional strategy

In this section we consider degenerate games in which there is only one edge out of each vertex for player II, say. This corresponds, for example, to cases in which player I knows in advance the positional strategy according to which player II is going to play. An $O(|V| \cdot |E|)$ algorithm of Karp [Kar78] (see also [CLR90], p. 548) for finding the maximum (or minimum) mean weight cycle of a weighted graph $G = (V, E)$ supplies, almost immediately, an efficient purely combinatorial algorithm for such special cases.

Theorem 6. *Let $G = (V, E)$ be a directed bipartite graph with a real weight function $w : E \rightarrow R$ on its edges, and assume that the outdegree of each vertex $v_2 \in V_2$ is exactly one. Then, the values of all the vertices and a positional optimal strategy $\pi_1 : V_1 \rightarrow V_2$ for player I can be found in $O(|V| \cdot |E|)$ time.*

Could Karp's algorithm be used to obtain a more efficient algorithm for the general case? The natural decision problem corresponding to MPG's is the following. Given an MPG G and a number ν, is the value of G at least ν? As a Corollary to Theorem 6 we get the following result.

Theorem 7. *The decision problem corresponding to MPG's is in NP $\cap$ co-NP.*

The simple observations of this section were first made by Gurvich, Karzanov and Khachiyan [GKK88], and by Karzanov and Lebedev [KL93]. They are included here for completeness.

5 Discounted payoff games

In this section we describe a *discounted* version of mean payoff games. This (fourth) variant, which is also interesting in its own right, will serve in the next section as a link between mean payoff games and simple stochastic games.

Let $0 < \lambda < 1$ be a real number. The weight of the i^{th} edge, e_i, chosen by the players is now multiplied by $(1-\lambda)\lambda^i$ and the outcome of the game is defined to be $(1-\lambda)\sum_{i=0}^{\infty}\lambda^i w(e_i)$. The goal of the first player is again to maximise, and of the second player to minimise, this outcome. The number λ is called the *discounting factor* of the game.

Let $G = (V_1, V_2, E)$ be a directed bipartite graph, where $V_1 = \{u_1, \ldots, u_{n_1}\}$ and $V_2 = \{v_1, \ldots, v_{n_2}\}$. Let a_{ij} be the weight of the edge (u_i, v_j), or $-\infty$ if $(u_i, v_j) \notin E$. Similarly, let b_{ji} be the weight of the edge (v_j, u_i), or $+\infty$ if $(v_j, u_i) \notin E$. Let $x_i = x_i(\lambda)$ be the value of a discounted game started at u_i and let $y_j = y_j(\lambda)$ be the value of a discounted game started at v_j. The following theorem is easily verified. Its proof is omitted.

Theorem 8. *The value vectors $(x_1, \ldots, x_{n_1})$ and $(y_1, \ldots, y_{n_2})$ of the discounted games played on $G = (V_1, V_2, E)$ are the unique solutions of the equations:*

$$\begin{aligned} x_i &= \max\nolimits_{1\le j\le n_2}\{(1-\lambda)a_{ij} + \lambda y_j\} \text{ for } 1 \le i \le n_1\ , \\ y_j &= \min\nolimits_{1\le i\le n_1}\{(1-\lambda)b_{ji} + \lambda x_i\} \text{ for } 1 \le j \le n_2\ . \end{aligned}$$

It follows immediately from this theorem that both players of the discounted game again have positional optimal strategies. Let $\nu(\lambda)$ be the value of the discounted game with discounting factor λ. As λ tends to 1, we expect $\nu(\lambda)$ to tend to ν, the value of the non-discounted game. This follows from the next theorem whose proof is omitted due to lack of space.

Theorem 9. *Let $G = (V, E)$ be a graph on n vertices, let $w : E \to \{-W, \ldots, 0, \ldots, W\}$ be a weight function on its edges and let λ be a real number satisfying $0 < \lambda < 1$. Let $\nu(\lambda)$ and ν be the values of the discounted and mean payoff games played on the graph $G = (V, E)$ starting at $a \in V$. Then,*

$$\nu - 2n(1-\lambda)W \ \le\ \nu(\lambda)\ \le\ \nu + 2n(1-\lambda)W\ .$$

In particular, if we choose $\lambda = 1 - 1/(2n^3W)$ then it is easy to verify that $|\nu(\lambda) - \nu| \le 1/(n(n-2))$, and ν can be obtained from $\nu(\lambda)$ by rounding to the nearest rational with an even denominator of at most n, as was done in Section 2. We thus obtain a reduction from MPG's to discounted payoff games (DPG's).

6 Reduction to simple stochastic games

In this section we describe a simple polynomial reduction from discounted payoff games (DPG's) to simple stochastic games (SSG's). This reduction, combined with the reduction from MPG's to DPG's, shows that SSG's are at least as hard as MPG's. We believe that MPG's are in fact easier than SSG's.

A *simple stochastic game* is a two-person game played on a directed graph $G = (V, E)$ whose vertex set V is the union of three disjoint sets V_{max}, V_{min} and V_{average}. The graph also contains a special start vertex and two special vertices called the 0-sink and the 1-sink. Each edge emanating from an 'average' vertex has a rational probability attached to it. The probabilities attached to all the edges from each average vertex add up to 1.

A token is initially placed on the start vertex of the graph. At each step of the game the token is moved from a vertex to one of its neighbours, according to the following rules:

1. At a max vertex, player I chooses the edge along which the token is moved.
2. At a min vertex, player II chooses the edge.
3. At an average vertex, the edge along which the token is moved is chosen randomly according to the probabilities attached to the outgoing edges.

The game ends when the token reaches one of the sink vertices. Player I wins if the token reaches the 1-sink and player II wins otherwise, i.e., if the token reaches the 0-sink or if the game does not end. The *value* of such a game is the probability that player I wins the game when both players play optimally. As was the case for mean payoff games, the two players of a simple stochastic game have positional optimal strategies.

Simple stochastic games were first studied by Shapley [Sha53]. Many variants of them have been studied since then (see Peters and Vrieze [PV87] for a survey). Condon [Con92] was the first to study simple stochastic games from a complexity theory point of view. She showed that the natural decision problem corresponding to SSG's (i.e., given a game G and a rational number $0 < \alpha \leq 1$, is the value of G at least α?) is in NP $\cap$ co-NP. No polynomial time algorithm for SSG's is yet known. A subexponential randomised algorithm for SSG's was obtained by Ludwig [Lud95].

Condon [Con92] actually shows containment in NP $\cap$ co-NP of the decision problem that corresponds to SSG's of the following restricted form. The outdegree of each non-sink vertex is exactly two and the probability attached to each edge that emanates from an average vertex is $1/2$. She then describes a reduction from general SSG's to SSG's of this restricted form. Her reduction, however, is not polynomial. A general SSG on n vertices in which the denominators of all the (rational) probabilities are at most m is transformed into a restricted SSG of size polynomial in n and m, rather than in n and $\log m$. Her transformation can be easily modified however, as we show next, to yield a polynomial reduction.

It is easy to transform a SSG into an equivalent SSG in which the outdegree of each non-sink vertex is exactly two. Each vertex of fan-out k is simply replaced by a binary tree with k leaves. This increases the size of the graph

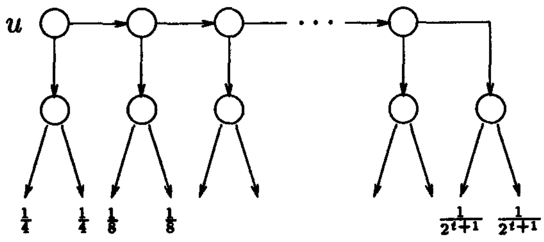

Fig. 1. Implementing an average vertex with arbitrary probabilities

(i.e., the number of vertices and edges) by only a constant factor. The remaining problem is therefore the simulation of binary average vertices with non-equal probabilities. Suppose we want to implement an average vertex u with two emanating edges (u, v_1) and (u, v_2), labeled respectively by the probabilities p/q and $(q-p)/q$, where p and q are integers and $2^{t-1} \le q < 2^t$. Let $a_1a_2\ldots a_{t-1}a_t$ and $b_1b_2\ldots b_{t-1}b_t$ be the binary representations of p and $q-p$ respectively, where a_1 and b_1 are the most significant digits. We use the construct shown in Figure 1. All the vertices used are average vertices with equal probabilities. For every $2 \le i \le t+1$, there are two emanating edges that are reached from u with probability 2^{-i}. If $a_i = 1$ then connect one of the edges which has probability $2^{-(i+1)}$ to v_1, and if $b_i = 1$ then connect one of these edges to v_2. All the unused edges are connected back to u. Is it easy to check that v_1 and v_2 are eventually reached with the appropriate probabilities. The number of vertices used in this construction is proportional to the number of bits needed to represent the transition probabilities. The reduction is therefore polynomial.

A simple stochastic game is said to *halt with probability 1* if, no matter how the players play, the game ends with probability 1. The proof of the following theorem can be found in Condon [Con92]. Note its similarity to Theorem 8.

Theorem 10. *Let $G = (V, E)$ be an SSG that halts with probability 1, and let $p(u, v)$ denote the probability attached to an edge (u, v) that emanates from an average vertex u. The values $\nu(v)$ of the vertices of G form the unique solution to the following set of equations:*

$$\nu(u) = \begin{cases} \max_{(u,v)\in E}\{\nu(v)\} & \textit{if } u \textit{ is a max vertex,} \\ \min_{(u,v)\in E}\{\nu(v)\} & \textit{if } u \textit{ is a min vertex,} \\ \sum_{(u,v)\in E}\{p(u,v)\cdot\nu(v)\} & \textit{if } u \textit{ is an average vertex,} \end{cases}$$

along with the conditions that $\nu(\text{0-sink}) = 0$ *and* $\nu(\text{1-sink}) = 1$.

We are finally in a position to describe a reduction from discounted payoff games (DPG's) to simple stochastic games (SSG's). Recall that we have already described a reduction from MPG's to DPG's.

Let $G = (V, E)$ be a DPG with discounting factor λ. If we add a constant c to all the weights of the game, the value of the game is increased by c. If we multiply all the weights of the game by a constant $c > 0$, the value of the game is multiplied by c. We can therefore scale the weights so that they will all be rational numbers in the interval $[0, 1]$. If the original weights were in the

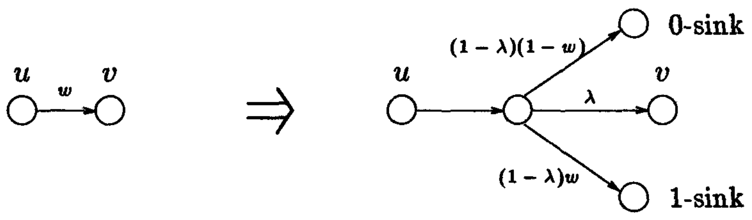

Fig. 2. Simulating a transition of a discounted payoff game.

range $\{-W, \ldots, 0, \ldots, W\}$, then the new weights will be rational numbers with denominators and numerators in the range $\{0, 1, \ldots, 2W\}$.

We construct an SSG $G' = (V', E')$, with the same value as the scaled DPG $G = (V, E)$ with discounting factor λ, in the following way. Each edge (u, v) with weight w in G is replaced by the construct shown in Figure 2. The simple stochastic game G' halts with probability 1, as in each transition there is a probability of $1 - \lambda$ of reaching a sink vertex. The values of the vertices of the discounted payoff game G satisfy the set of equations given in Theorem 8. The values of the vertices of the simple stochastic game G' satisfy the set of equations given in Theorem 10. These two sets of equations become *identical* once the intermediate variables, that correspond to the intermediate vertices introduced by the transformation described in Figure 2, are eliminated. As this set of equations has a unique solution, the values of the two games are equal. The transformation of G to G' can clearly be carried out in polynomial time. This completes the description of the reduction.

7 Some applications

In this section we briefly mention two applications of mean payoff games.

Consider a system that can be in one of n possible states. At each time unit, the system receives one of k possible requests. The system is allowed to change its state and then it has to serve the request. The transition from state i to state j costs a_{ij}, and serving a request of type t from state i costs b_{it}. What, in the worst case, is the average cost of serving a request?

Borodin, Linial and Saks [BLS92] performed a *competitive analysis* of such systems, which they call *on-line metrical task systems*. If we look at the worst case instead, we get a mean payoff game played between the system and an adversary that chooses the requests.

Consider finite-window on-line string matching algorithms (see [CHPZ95] for a definition). What, in the worst case, is the average number of comparisons that an optimal algorithm has to perform per text character? The problem can be formulated as a mean payoff game played between the designer of a string matching algorithm and an adversary that answers the queries made by an algorithm. The reward (the complement of cost) obtained by the algorithm at each stage is the amount by which it can shift its window. For each pattern string and window size we obtain a mean payoff game, the solution of which yields an optimal string matching algorithm for that pattern and window size.

8 Concluding remarks

Mean payoff games form a very natural class of full-information games and we think that resolving their complexity is an interesting issue. We conjecture that they lie in P but, since none of the standard methods seems to yield a polynomial time algorithm for them, the study of mean payoff games may require new algorithmic techniques. If such positive approaches are unsuccessful, the example of mean payoff games may help in exploring the structure of NP $\cap$ co-NP.

Acknowledgments

We would like to thank Sergiu Hart, Ehud Lehrer, Nimrod Megiddo, Moni Naor, Noam Nisan and Avi Wigderson for helpful discussions and suggestions, Alexander Karzanov for pointing out a flaw in the previous version of Theorem 5, and Thomas McCormick for bringing references [GKK88] and [KL93] to our attention.

References

[BLS92] A. Borodin, N. Linial, and M.E. Saks. An optimal on-line algorithm for metrical task system. *Journal of the ACM*, 39(4):745–763, 1992.

[CHPZ95] R. Cole, R. Hariharan, M. Paterson, and U. Zwick. Tighter lower bounds on the exact complexity of string matching. *SIAM Journal on Computing*, 24:30–45, 1995.

[CLR90] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to algorithms*. The MIT Press, 1990.

[Con92] A. Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, February 1992.

[EM79] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8:109–113, 1979.

[GKK88] V.A. Gurvich, A.V. Karzanov, and L.G. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *USSR Computational Mathematics and Mathematical Physics*, 28:85–91, 1988.

[Kar78] R.M. Karp. A characterization of the minimum cycle mean in a digraph. *Discrete Mathematics*, 23:309–311, 1978.

[KL93] A.V. Karzanov and V.N. Lebedev. Cyclical games with prohibitions. *Mathematical Programming*, 60:277–293, 1993.

[Lud95] W. Ludwig. A subexponential randomized algorithm for the simple stochastic game problem. *Information and Computation*, 117:151–155, 1995.

[Pra75] V.R. Pratt. Every prime has a succinct certificate. *SIAM Journal on Computing*, 4(3):214–220, 1975.

[PV87] H.J.M. Peters and O.J. Vrieze. Surveys in game theory and related topics. CWI Tract 39, Centrum voor Wiskunde en Informatica, Amsterdam, 1987.

[Sha53] L.S. Shapley. Stochastic games. *Proc. Nat. Acad. Sci. U.S.A.*, 39:1095–1100, 1953.