Linear Programming Polytope and Algorithm for Mean Payoff Games*

Ola Svensson[1,**] and Sergei Vorobyov[2]

[1] IDSIA, Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, Lugano, Switzerland
[2] Information Technology Department, Uppsala University, Sweden
Sergei.Vorobyov@it.uu.se

Abstract. We investigate LP-polytopes generated by mean payoff games and their properties, including the existence of tight feasible solutions of bounded size. We suggest a new associated algorithm solving a linear program and transforming its solution into a solution of the game.

1 Introduction

The goal of this paper is to investigate linear programming formulations for *mean payoff games* (MPGs) [8, 9], a well-known problem in $\mathrm{NP} \cap \mathrm{coNP}$, with an open P-membership status. Recently combinatorial randomized subexponential algorithms for linear programming were successfully applied for solving several kinds of games [5, 6, 2, 1, 3]. However, to our knowledge, there are no previous attempts at investigating LP-formulations for MPGs and associated polyhedra, except recent work [2, 1, 3] representing some infinite games as instances of the new so-called *controlled linear programming problem (CLPP)*. In contrast, LP-formulations and relaxations are well studied and understood for the overwhelming majority of combinatorial optimization problems [13].

Several naturally arising questions we address and solve are as follows.

1. Is it possible to describe/approximate solutions to an MPG by linear constraints, i.e., as a polyhedron or a polytope?
2. Do "real" solutions to MPGs lie inside this polytope, how can they be characterized, are they vertices of this polytope?

We present several surprisingly interesting and simple properties, classifying feasible solutions of the MPG-polytopes and giving new insights into the combinatorial structure of the problem. Based on these, we describe a new MPG-solving algorithm, which solves a linear program and then transforms (if necessary) an optimal solution into a solution of the game by "tightening".

* Research supported by the grants from the Swedish Scientific Council and the Foundation for International Cooperation in Research and Higher Education.
** Partially supported by the Swiss National Science Foundation Project 200020-109854.

S.-W. Cheng and C.K. Poon (Eds.): AAIM 2006, LNCS 4041, pp. 64–78, 2006.

More specifically, we represent an MPG by a linear system with a *totally unimodular* matrix, defining a nonempty integral MPG-polytope. Some vertices of this polytope represent the so-called "*tight*" feasible solutions, which solve the corresponding MPG. We suggest a new algorithm for finding tight solutions based on minimizing a simple linear function and "tightening" an optimal solution. In contrast to the TSP-polytope, for which no LP-description is to be expected, MPG-polytopes are easily characterized, and any game can be solved by optimizing a single (but unknown) linear function over such polytope. This provides for a certain reduction in the size of the search space and is suggestive for a new class of algorithms.

Combinatorial optimization and linear programming seem to be very productive tools for solving games. In [6] we generalized the *shortest paths* problem to the *controlled* or *longest* shortest paths problem, and used it together with combinatorial linear programming for solving MPGs. Combinatorial structures underlying iterative improvement for games are explored in [5]. A related line of research concerns applications of the *Linear Complementarity Problem* (LCP) [10, 7], a nonlinear optimization theory we recently successfully applied to solving several classes of infinite games and P-matrix Generalized LCPs [4, 14].

2 Preliminaries

2.1 Mean Payoff Games

We start by recalling basic definitions about mean payoff games (MPGs) and then introduce the 0-mean partition problem, to which all other problems for MPGs are polynomially reducible. The 0-mean partition problem is convenient for the linear programming formulations and simplifies descriptions of different algorithms. We further show that simplifying restrictions to ergodic MPGs (all vertices have the same value), ergodic bipartite MPGs (players strictly alternate moves), and ergodic complete bipartite MPGs (the game graph is complete bipartite) can be done without loss of generality.

A *mean payoff game (MPG)* is a two-player game, played on a finite directed edge-weighted graph $G = (V, E, w)$, where the set of vertices V is partitioned into two nonempty sets $V_{\max}$, $V_{\min}$, every vertex has at least one outgoing edge (no sinks or leaves), and the weight function w is integer-valued.

We assume throughout the paper that $n = |V|$ is the number of vertices of the game graph G, $n_{\max} = |V_{\max}|$, $n_{\min} = |V_{\min}|$, and W is the maximal absolute edge weight; thus $w : E \to \{-W, \dots, W\}$.

Given an MPG, a play develops in the following way. Initially, a pebble is placed in some vertex v_0 and players MAX and MIN start constructing an infinite sequence of edges $\{(v_i, v_{i+1})\}_{i=0}^{+\infty}$. If the pebble is in a vertex $v_i \in V_{\max}$ then MAX selects an outgoing edge from v_i and moves the pebble to its destination vertex v_{i+1}, otherwise MIN makes the analogous choice and move.

Players MAX and MIN are adversaries, the first one wants to maximize, whereas the second one wants to minimize, respectively, the values

$$\liminf_{k\to\infty} \frac{1}{k}\sum_{i=0}^{k-1} w(v_i, v_{i+1}), \quad \text{and } \limsup_{k\to\infty} \frac{1}{k}\sum_{i=0}^{k-1} w(v_i, v_{i+1}). \tag{1}$$

It turns out that MPGs are solvable in *pure positional* strategies for both players, and every vertex has a *value* $\nu(v)$ [8, 9]. This value is equal to both limits in (1), and both players can secure it by applying these strategies. Moreover, when one player fixes his pure positional strategy, an optimal counterstrategy of his adversary is polynomial time computable. Consequently the problem whether the value of a vertex is above/below a certain threshold is in NP $\cap$ coNP.

An MPG is called *bipartite* if $E \subseteq (V_{\max} \times V_{\min}) \cup (V_{\min} \times V_{\max})$, i.e., players strictly alternate moves. A bipartite MPG is *complete* if $E = (V_{\max} \times V_{\min}) \cup (V_{\min} \times V_{\max})$, and *incomplete* otherwise. As usual, the weight of a cycle is the sum of edge weights along the cycle.

2.2 0-Mean Partition Problem for MPGs

In this paper we concentrate on the following restricted problem, which polynomially subsumes the problem of computing values of MPGs (as well as other problems, as ergodic partitioning, finding optimal strategies). It also simplifies the the algorithms, structure and properties of LP-representations.

0-Mean Partition Problem for MPGs.

Given: a bipartite MPG G without 0-weight cycles.
Find: a partition of vertices of G into sets $G_{>0}$ and $G_{\leq 0}$ of vertices with positive and nonpositive values. □

Restricting to this problem, with the additional constraints as stated, is no loss of generality. We summarize it in the following two propositions.

Proposition 1. *Finding values of MPGs is polynomial time reducible to the 0-mean partitioning problem.*

Proof. For an arbitrary MPG, adding a constant k to every edge weight adds k to every vertex value; multiplying every edge weight by a constant k multiplies every vertex value by k. This is because values are defined by mean values of optimal cycles wrt positional strategies, and because every cycle mean changes by additive or multiplicative constant, respectively. Therefore, partitioning with a rational mean threshold reduces to 0-mean partitioning.

Values of an MPG vertices are rationals with numerators and denominators up to nW and n, respectively. If a value is known to belong to an interval of length $\leq 1/n^2$, then it is uniquely determined (the smallest difference between two values is $\frac{1}{n-1} - \frac{1}{n}$). Bisecting the range $[-W, W]$ with rational thresholds, polynomially many in n and $\log W$ times, each time invoking the partition algorithm, we may uniquely determine the value of a vertex [9, 15, 6]. □

Proposition 2. *In the 0-mean partition problem the following assumptions can be done without loss of generality:*

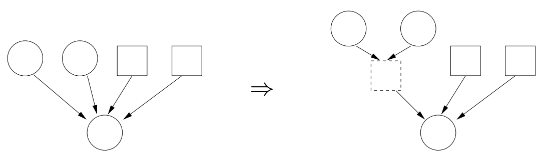
1. the game graph has no 0-weight cycles,
2. the game graph is bipartite,
3. the values of all vertices are of the same sign;
4. the game graph is complete bipartite (with $|V_{\max}| = |V_{\min}|$).

Every reduction from general MPGs to a restricted case is polynomial.

Proof. Given an arbitrary MPG, consider the following chain of reductions.

1. Multiplying every edge weight by $n+1$ and subtracting one does not change signs of positive- and negative-weight cycles, but 0-weight cycles (if any) become negative-weight. The 0-mean partition remains the same.

2. The straightforward solution is to introduce a vertex of the opposite player between two vertices of the same player. This, however, may increase the number of vertices quadratically. A more economic solution, leading to just a linear increase in the number of vertices is depicted in the figure below (the outgoing edge from the new vertex gets weight 0). Note that this transformation may actually change means of cycles, but not 0-means partitions, which is enough for our purpose of computing values.



3. Let v be an arbitrary vertex of a bipartite MPG G without 0-weight cycles. Construct G' by adding a new backward edge from every vertex $u \neq v$ of G to v of weight $-M$ for an edge from a Max vertex and of weight $+M$ for a Min vertex, where $M = (n-1)W + 1$. Suppose, a play in G' starts from v. If Max can secure a positive value of v in G, he can use the same strategy as in G never using new edges. If Min never uses his new edges, then the value is the same as in G. But if Min is the first to use his heavy edge back to v, the cycle thus formed has a mean $\geq [(n-1)W + 1 - (n-1)W]/n > 0$ (and we refer to the equivalence of finite and infinite MPGs [8]). The case when v has negative value in G is symmetric. Now suppose a play starts in any other vertex $v' \neq v$. Then each player can reach v and then follow the same strategy as he uses from v. The signs of means in *infinite* plays thus formed, one starting from v', the other from v, are the same (the initial finite path does not matter (this argument also depends on the equivalence between finite and infinite MPGs [8]). Hence, in G' values of all vertices are of the same sign.

4. Let $M = (n-1)W + 1$. Add all missing edges between $V_{\max}$ and $V_{\min}$ of weight $-M$ or $+M$ depending on whether an edge leaves a Max or a Min vertex. This makes the graph complete bipartite and preserves the signs of all values. We can also assume both partitions have the same number of vertices. □

Remark 1. In the above chain of reductions the numbers of vertices and edges grow just linearly in the number of vertices n. In contrast, maximum absolute weights in each of 1, 3, and 4 are multiplied by n, resulting in the overall weight multiplication by n^3. Our algorithm operates on bipartite MPGs without 0-weight cycles. Thus, assumptions 1, 2 cost us a factor of n in the weight increase.

2.3 Longest Shortest Paths (LSP)

The Longest Shortest Paths problem has previously been successfully applied to solve MPGs in *randomized subexponential* time [6, 5]. Here we will use it to prove the existence of small tight feasible solutions of the MPG-generated systems of linear constraints (Section 4).

THE LONGEST SHORTEST PATH PROBLEM.

Given: a weighted digraph (without 0-weight cycles) with a sink and a set of *controlled* vertices.

Find: a selection of *exactly one* edge from each controlled vertex maximizing the lengths of the shortest paths from each vertex to the sink. □

3 LP Formulations for MPGs

The definition below does not assume that an MPG is bipartite nor complete.

Definition 1 (Linear Slack Constraints). *For an MPG G let S_G, called* slack constraints*, be the following system of linear constraints:*

1. *for every edge $x \xrightarrow{w} v$ with $x \in V_{\max}$ write constraints*
$$x = v + w_{xv} + s_{xv}, \tag{2}$$
$$s_{xv} \geq 0, \tag{3}$$
where s_{xv} is a MAX slack variable *for the edge;*
2. *similarly, for every edge $y \xrightarrow{w} v$ with $y \in V_{\min}$ write constraints*
$$y + s'_{yv} = v + w'_{yv}, \tag{4}$$
$$s'_{yv} \geq 0, \tag{5}$$
where s'_{yv} is a MIN slack variable *for the edge.* □

We adopt the convention that primed s' and w' denote MIN slacks and weights of edges outgoing from MIN vertices. In the sequel we will freely identify edges with their corresponding equality constraints.

The simple LP-formulation above allows one to derive many interesting MPG properties to be discussed below. We start with the simplest, but useful

Proposition 3. *For a cycle in an MPG G let w_i, s_i, s'_i (for $i \in I$) be weights of all edges, all* MAX *slacks, and all* MIN *slacks on the cycle. Then S_G implies*
$$\sum_{i\in I} w_i + \sum_{i\in I} s_i - \sum_{i\in I} s'_i = 0. \tag{6}$$

Proof. Just sum up left- and right-hand sides of the equalities corresponding to edges on the cycle. □

This proposition partially explains why the bipartite requirement is useful. Indeed, whenever a positive weight cycle traverses only MAX vertices in G, or a negative weight cycle traverses only MIN vertices, the system S_G is infeasible, because (6) cannot be satisfied.

With the introductory purpose of explaining the usefulness of linear slack constraints, let us temporarily assume complete bipartiteness. Say that a solution to a linear slack system is *tight for* MAX (for MIN, resp.), if for every MAX vertex (MIN vertex, resp.) at least one outgoing edge has slack zero (we call such edges *tight*). The following proposition shows that tight solutions determine the winner.

Proposition 4. *If the system of slack constraints has a tight solution for*

1. MAX*, then* MAX *can enforce a nonnegative cycle in the corresponding MPG from every vertex;*
2. MIN*, then* MIN *can enforce a nonpositive cycle in the corresponding MPG from every vertex.*

Proof. Let MAX use any tight edges with zero slacks as his strategy. Then, by (6), for every cycle that MIN can create the sum of edge weights on the cycle is nonnegative. The proof of the second claim is analogous. □

The next section addresses the existence of (tight) solutions for the MPG linear slack constraints and their relation to determining the winner. Now we introduce MPG-polyhedra.

Definition 2 (MPG Polyhedron). *An MPG-polyhedron is the feasible set of the linear slack constraints corresponding to an MPG; see Definition 1.* □

We have seen above that some MPGs may induce empty polyhedra. The next section shows that bipartite MPGs always have nonempty polyhedra. Here we state simple properties of MPG-polyhedra.

Proposition 5. *An MPG-polyhedron has no vertices.*

Proof. Suppose (x, y, s) is a vertex. Then $(x + \alpha 1, y + \alpha 1, s)$ (1 is a vector of ones, $\alpha \in \mathbb{R}$) is also a feasible solution to slack constraints. Thus, any MPG-polyhedron, with each point contains a line, hence has no vertices. □

In Section 4 we introduce additional bounding constraints and an MPG-polyhedron becomes an MPG-polytope (bounded polyhedron), with vertices.

Another useful property of MPG-polyhedra is their integrality.

Proposition 6. *For any MPG-polyhedron P one has $conv(P) = conv(P_I)$.*

Proof. Any MPG-generated linear slack system can be written as $[A\ I](x, y, s)^T = b$, where the entries in A correspond to x and y variables, and the identity matrix corresponds to the slacks. Every row of A has *exactly* one $+1$ and one -1 entry and is thus totally unimodular. Totally unimodularity for $[A\ I]$ follows directly, since it is preserved when adding a column with at most one nonzero, being ± 1 [12, p. 280]. By [12, Theorem 19.1, p. 266], the polyhedron $\{v|\,[A\ I]\,v \leq b\}$ is integral whenever b is integral. Duplicating a row and multiplying a row by -1 preserve total unimodularity and the polyhedron $\{v|\,[A\ I]\,v = b\}$ is integral. □

As a consequence, any linear function over an MPG-polyhedron with finite optimum, has an integral optimum. Moreover, optimizing any linear function over an MPG-polyhedron can be done in *strongly* polynomial time, because the constraint matrix consists of 0 and ± 1 entries.

4 Existence of Tight Solutions

In this section we consider linear slack systems corresponding to bipartite (not necessarily complete) MPGs and show that they possess tight feasible solutions of bounded size. We first generalize the notion of tightness, introduced (for the case of complete bipartite MPGs) in the previous section.

Definition 3 (Tight Solution). *Given a linear slack system S_G obtained from a bipartite MPG G without 0-weight cycles, say that a solution to S_G is* tight *if there is a partition of vertices of G into sets X and N such that:*

1. *every* Min *vertex in X has a tight edge to X;*
2. *every* Max *edge from X leads to X;*
3. *every* Max *vertex in N has a tight edge to N;*
4. *every* Min *edge from N leads to N.* □

(Note that in the case of an ergodic MPG, e.g., a complete bipartite MPG, either X or N should be necessarily empty.)

A tight solution to a slack system gives the 0-mean partitioning for the associated MPG as shows the following

Proposition 7. *$G_{>0} = N$ and $G_{\leq 0} = X$.*

Proof. If a play starts in N, then Max may just use his tight edges to stay in N. When a cycle is eventually formed, by (6), the sum of weights on the cycle is positive (there are no 0-weight cycles); hence, the mean is also positive.

Symmetrically, if a play starts in X, then Min just uses his tight edges to stay in X. When a cycle is eventually formed, by (6), the sum of weights on the cycle is nonpositive; hence, the mean is also nonpositive. □

Here comes the main result of this section. Although there are well-known general bounds on some feasible solution to a system of linear constraints (if it exists) [12, Ch. 10], our bounds for MPG-generated constraints are stronger. We also show that tight solutions of bounded size always exist. In Section 5 we prove related results for complete bipartite MPGs.

Theorem 1 (Tight Solution Existence). *A linear slack system of every bipartite MPG without 0-weight cycles always has a tight solution with integral components of absolute value $O(nW)$, where n is the number of vertices and W is the maximal absolute edge weight.*

Proof. Add *retreat* edges, of weight 0, from all Max vertices to the sink (new vertex), and of weight $M = (2n-1)W + 1$ from all Min vertices to the sink. The resulting graph determines an instance of the *Longest Shortest Paths* (LSP) problem [6]. In this instance optimal positional strategies of both players create no cycles, because each cycle is either positive or negative, which one of the players always wants to avoid (and can due to bipartiteness). Thus all optimal plays end up in the sink, through a 0- or M-weight retreat edge. The *unique* [6] solution (with all components finite, because every cycle is broken by one of the players selecting to retreat) determines a feasible solution to the linear slack system. Optimal edges for both players have associated slacks equal zero. Moreover, by the properties of the shortest paths [6] and optimality for both players, the following conditions are satisfied for every edge (v, u) of the game graph, because $d(v)$, $d(u)$ are shortest path distances:

$$d(v) \leq w(v,u) + d(u), \text{ if } v \in V_{\min}, \quad (7)$$

$$d(v) \geq w(v,u) + d(u), \text{ if } v \in V_{\max}. \quad (8)$$

These conditions ensure that all slacks are *nonnegative*. Moreover, at least one slack per vertex is zero, since $d(v)$ are defined by shortest paths.

Let the required sets X and N be as follows:

1. N is the set of vertices starting from which Max can force a play into a Min vertex from which Min retreats through the retreat edge with weight M, when both players can use tight edges only;
2. X is the set of vertices starting from which Min can force a play into a Max vertex from which Max retreats through the retreat edge with weight 0, when both players can use tight edges only.

The graph on tight edges is acyclic, bipartite, spanning all vertices of the game graph, with leaves being vertices selecting retreat edges. Therefore, N and X form a partition, which can be easily computed, after topological sorting, by dynamic programming. We have to show that Max has no edges (including non-tight) from X to N and Min has no edges (including non-tight) from N to X (see Definition 3).

Since X and N do not intersect and shortest distances inside them are defined by tight edges, the choice of the weights for the retreat edges implies the bounds on the values of Max and Min vertices in X and N summarized in the table.

	X	N
Max	$[0, (n-1)W]$	$[nW+1, 2nW+1]$
Min	$[-W, (n-1)W]$	$[nW+1, (2n-1)W+1]$

In the left column, the common upper bound is explained by the fact that the longest path in X may traverse at most $n-1$ edges of weight at most W. The lower bounds 0 and $-W$ in the left column are due to the MAX retreat and to bipartiteness: the best MIN can do is to go to the 0-value vertex via a $-W$ edge. In the right column, the common lower bound is because the shortest path in N is through the M-weighted retreat and at most $n-1$ edges of weight $-W$. The upper bound for a MIN variable is due to the retreat weight, and for a MAX variable it is just W larger.

To show that MAX has no edges from X to N, assume, toward a contradiction, that MAX has an edge from $v \in X$ to $u \in N$. The bound from the table above together with (8) imply $w(v,u) < -W$, a contradiction, since W is the maximal absolute edge weight. A similar argument shows that MIN cannot have edges from N to X.

Now delete the sink and retreat edges to return to the original game. All equalities in the associated linear slack system are satisfied. This solution is tight as shown above. Note that some 0 slacks for some variables can disappear (in the vertices where a retreat was taken).

Since a slack s is always equal $s = x - y \pm w$, from the table above we conclude that all slacks are at most $O(nW)$. □

Remark 2. We can thus impose additional bounding constraints for all variables in the linear slack systems from Definition 1. The feasible set becomes a polytope with vertices, which we call an *MPG-polytope.*

Proposition 8. *An MPG-polytope of a bipartite game always has at least one vertex, which is a tight solution.*

Proof. Consider a tight solution, which exists by Theorem 7. Minimize the sum of slacks, which are zero in the tight solution, over the MPG-polytope. Obviously, the value of the optimum will be zero. Furthermore, the optimal solution can be attained in a vertex of the polytope. □

Proposition 11 shows a simple form of a linear target function for a complete bipartite MPG with an optimum attained in a tight solution.

Corollary 1. *Vertices of an MPG-polytope of a bipartite game are integral.* □

5 MPGs on Complete Bipartite Graphs

In this section we assume that MPGs are played on complete bipartite graphs $K_{p,p}$. Thus the number of vertices $n = 2p$. We use a convention that x_i, y_i denote variables associated to the i-th vertex of MAX and MIN respectively, s_{ij} and s'_{ij} denote slacks for MAX and MIN edges, and w_{ij}, w'_{ij} denote edge weights of MAX and MIN. Slack equality constraints (2) and (4) in this case are (for $1 \leq i, j \leq p$):

$$x_i = y_j + w_{ij} + s_{ij}, \tag{9}$$
$$y_i + s'_{ij} = x_j + w'_{ij}. \tag{10}$$

5.1 Invariant Properties

Proposition 9. *Every solution to a linear slack constraint system S_G obtained from a complete bipartite MPG G satisfies the invariant*

$$\sum_{ij} s_{ij} - \sum_{ij} s'_{ij} = -\sum_{ij}(w_{ij} + w'_{ij}).$$

Proof. Sum up all equalities (9) and (10). This gives $\sum s'_{ij} = \sum s_{ij} + \sum(w_{ij} + w'_{ij})$, since each variable x_i, y_i appears in the left- and right-hand sides of (9), (10) the same number of times. □

The following proposition shows that one can optimize any of the several linear functions over the MPG-polytope. They happen to possess the same optimal solutions, i.e., are equivalent.

Proposition 10. *For any complete MPG-generated S_G the following functions are similar up to scaling and a constant additive term:*

$$1) \sum_{i,j} s_{ij}, \quad 2) \sum_{i,j} s'_{ij}, \quad 3) \sum_{i,j} s_{ij} + \sum_{i,j} s'_{ij}, \quad 4) \sum_{i} x_i - \sum_{i} y_i.$$

Proof. Equivalence of 1-3 follows from Proposition 9. To prove equivalence of 1 and 4, we use the fact that $s_{ij} = x_i - y_j + w_{ij}$. Thus

$$\begin{aligned}
\textstyle\sum_{ij} s_{ij} &= (x_1 - y_1 + w_{11}) + (x_1 - y_2 + w_{12}) + \ldots + (x_1 - y_n + w_{1n}) + \\
&\quad (x_2 - y_1 + w_{21}) + (x_2 - y_2 + w_{22}) + \ldots + (x_2 - y_n + w_{2n}) + \\
&\quad \vdots \\
&\quad (x_n - y_1 + w_{n1}) + (x_n - y_2 + w_{n2}) + \ldots + (x_n - y_n + w_{nn}) \\
&= n(\textstyle\sum_i x_i - \sum_i y_i) + c, \text{ where } c \text{ is a constant.}
\end{aligned}$$ □

5.2 Complete Bipartite MPGs as Linear Programs

The next proposition asserts that there is always a simple linear target function over the feasible polytope of a complete bipartite MPG with the optimum, which solves the game.

Proposition 11. *Let S_G be a linear slack system obtained from a complete bipartite MPG. Then there exist vectors $a, b \in \mathbb{N}^p$ such that $\sum_i a_i = \sum_i b_i = p$ and the optimal solution to S_G with the objective function* $\min \sum_i a_i x_i - \sum_i b_i y_i$ *has either a tight solution for* Max *or for* Min *and thus solves the corresponding MPG. Moreover, one of the vectors a, b consists of ones only.*

Proof. Suppose Max has a winning strategy, hence a tight solution. Then the sum of the slacks corresponding to his optimal edges (tight), taken one per vertex, $\sum_{(i,j)\in I} s_{ij}$ has minimal solution 0. But this sum is equal $\sum_{(i,j)\in I}(x_i - y_j - w_{ij}) = \sum_{i-1}^{n} x_i - \sum_{j=1}^{n} b_j y_j + C$, where b_j counts how many times y_j is selected as a destination of some Max optimal edge. The proof, when Min has a winning strategy is symmetric. □

As a consequence, for a complete bipartite MPG, the corresponding slack polytope has a vertex solving the game (which also follows by Proposition 8). We state two other simple corollaries.

Corollary 2. *The problem of deciding the winner for a complete MPG reduces to the problem of determining:*

1. *the number of* Max *vertices that play, in a winning positional strategy, to the* Min *vertex y_i, for each i, if* Max *has a winning strategy, or*
2. *the number of* Min *vertices that play, in a winning positional strategy, to the* Max *vertex x_i, for each i, if* Min *has a winning strategy.* □

Corollary 3. *If* Max *has a winning strategy where every* Max *vertex selects an unique* Min *vertex. The game is solvable with the objective function* $\min \sum_i x_i - \sum_i y_i$. *The case for* Min *is symmetric.* □

5.3 Search Space

Proposition 11 allows one to somewhat reduce the search space of all positional strategies in a complete bipartite MPG.

Proposition 12. *The problem of finding vectors a, b such that it is possible to recover the winning player from the optimal solution to S_G with objective function* $\min \sum_i a_i x_i - \sum_i b_i y_i$ *has strictly smaller search space than deciding the optimal strategy of one player.*

Proof. In a complete MPG G played on the graph $K_{p,p}$ both players have p^p number of strategies.

Consider the problem of finding vectors a, b recovering the winning player from an optimal solution to S_G with objective function $\min \sum_i a_i x_i - \sum_i b_i y_i$ (as explained in the proof of Proposition 11).

If Max has a winning strategy, we can assume $a = 1$. It remains to find the correct b_i's. Any vector b with p nonnegative integer components summing up to p can be represented by a word of $p-1$ zeros (bucket separators) and p ones, i.e., p buckets and p items. The number of possible ways to distribute the items are $(2p-1)!/(p!(p-1)!) = \binom{2p-1}{p} = O(2^{2p})$.

Similarly, if Min is winning the number of ways to select the vector a is $O(2^{2p})$. Thus, the number of different meaningful objective functions are bounded by $O(2^{2p})$, which is $o(p^p) = o(2^{p \log p})$. □

6 0-In-Out Property

In this section we only assume that MPGs are bipartite, but not necessarily complete. Consider the following interesting

Definition 4 (0-in-out property). *Say that a solution to an MPG-generated system of slack constraints satisfies the* 0-in-out property *if*

$$\forall i \in V_{\max} \, \exists j \in V_{\min} (s_{ij} = 0 \vee s'_{ji} = 0) \wedge \forall i \in V_{\min} \, \exists k \in V_{\max} (s'_{ik} = 0 \vee s_{ki} = 0).$$

Informally, it stipulates that every vertex has at least one incoming or outgoing 0-slack (tight) edge. Two propositions below summarize interesting relations between tight solutions to systems of slack constraints, solutions minimizing $\sum x_i - \sum y_i$,[1] and solutions with the 0-in-out-property.

Proposition 13. *Every solution to an MPG-generated system of slack constraints, which minimizes $\sum x_i - \sum y_i$, possesses the 0-in-out property.*

Proof. An x_i with nonzero slacks on all outgoing and incoming edges can be decreased thus diminishing the target value. Similarly, a y_i with nonzero slacks on all outgoing and incoming edges can be increased thus diminishing the target value. □

Proposition 14. *For every* Max- *or* Min-*tight solution to an MPG-generated system of slack constraints there corresponds a tight solution satisfying the 0-in-out property with a smaller or equal target value $\sum x_i - \sum y_i$.*

Proof. A Max-tight solution has 0-in-out property satisfied for all Max vertices. If the property is not satisfied for a vertex y_i, then its value can be increased, keeping the tightness, and decreasing the target value. The proof for the Min-tight solutions is completely similar. □

6.1 Minimizing Slacks Does Not Give Tight Solutions

Thus, both: 1) tight solutions (modified, if necessary as explained in the proof of Proposition 14) and 2) solutions minimizing $\sum x_i - \sum y_i$, satisfy the 0-in-out property. A natural challenging question is: *whether tight solutions can always be found among minimizing $\sum x_i - \sum y_i$?* This plausible conjecture, if true, would allow us to limit the search for tight solutions among those minimizing $\sum x_i - \sum y_i$. Unfortunately, this promising conjecture fails, as demonstrated by the counterexample in Figure 1.

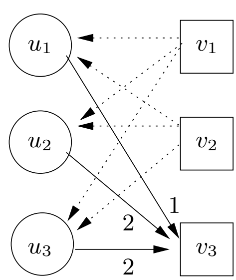


Fig. 1. A complete bipartite MPG where the dotted edges have weight −2 and the edges that are not in the figure have weight 0

[1] Recall that x_i, y_i are variables associated with the i-th vertex of Max and Min.

By Proposition 10, for any systems of linear slack constraints corresponding to complete bipartite MPGs, minimizing the objective function $\sum_i x_i - \sum_i y_i$ is equivalent to minimizing the objective functions $\sum\{$Max slacks$\}$, $\sum\{$Min slacks$\}$, and $\sum\{$All slacks$\}$.

It is easy to see that the value of $\min \sum_i x_i - \sum_i y_i$, when Max uses his winning strategy (always plays to v_3) is 7, because $u_1 = v_3 + 1$, $u_2 = v_3 + 2$, $u_3 = v_3 + 2$, $v_2 = u_1 - 2$, $v_1 = u_1 - 2$. Letting all Max variables equal 2 and all Min variables equal 0 is also feasible, but then u_1 has no tight outgoing edges. Thus, a Max- or Min-tight solution can have a larger value than the minimal value of the objective function $\sum_i x_i - \sum_i y_i$.

7 Slacks Update "Tightening" Algorithm

Despite the fact (described by the previous counterexample) that there may be no tight solutions (solving MPGs) among those minimizing $\sum x_i - \sum y_i$, the idea to start from such a solution and transform it into a tight one seems quite tempting. We now develop this idea and describe an algorithm for finding tight solutions for MPG-generated systems of slack constraints, and thus solves MPGs by Proposition 7. The algorithm applies to systems obtained from bipartite (not necessarily complete) MPGs without 0-weight cycles. The proof of correctness and the intuitions underlying the algorithm go in parallel with its description.

The Algorithm starts by finding a solution to slack constraints minimizing $\sum x_i - \sum y_i$ (in strongly polynomial time). By Proposition 13, every vertex has at least one (incoming or outgoing) tight edge.

Main Loop. Let X_0 and N_0 be the sets of Max and Min vertices without tight outgoing edges. If one of these sets is empty, the 0-mean partition is found (Proposition 7). Temporarily delete all non-tight edges. Let X be the set of vertices starting from which Min can force a play into X_0, and N be the set of vertices from which Max can force a play into N_0. (Both sets may be easily computed in polynomial time, as shown below.)

We claim that X and N form a partition of the game vertices. Indeed, every vertex is an endpoint (source or destination) of at least one tight edge. Note also that the graph induced by tight edges is acyclic (this follows from Proposition 3, because a cycle with all slacks 0 should be 0-weight, absent by assumption). Topologically sort it, and proceed from leaves (which are either in $X_0 \subseteq X$ or in $N_0 \subseteq N$) backwards in the topological order as follows. For a Max vertex v with all successors already decided to be in X or N, put v to N if it has a tight edge to N, and to X otherwise, and symmetrically for a Min vertex. This classifies all vertices as members of either X or N. At this stage:

- there are no tight Max edges from X to N, by definition of X; equivalently, all Max edges from X to N, denote them $E_{\max}(X, N)$, are non-tight;
- there are no tight Min edges from N to X, by definition of N; equivalently, all Min edges from N to X, denote them $E_{\min}(N, X)$, are non-tight;
- note that there may exists tight Max edges from N to X, as well as tight Min edges from X to N.

Terminate? If the set of edges $E_{\max}(X, N) \cup E_{\min}(N, X)$ is empty, the 0-mean partition is found: $G_{\leq 0} = X$ and $G_{>0} = N$ (see Proposition 7), and the algorithm terminates. (Both X, N may be nonempty if the graph is not complete bipartite.)

Update. Let $\delta > 0$ be the *minimal* slack assigned to edges in $E_{\max}(X, N) \cup E_{\min}(N, X)$ (all such edges are non-tight; see above). Now, either 1) *increase* the values of all vertices in N by δ, or 2) *decrease* the values of all vertices in X by δ. This does not violate any constraints, and preserves the property that every vertex has at least one in- or outgoing tight constraint/edge. Indeed, all constraints corresponding to edges from X to X and from N to N remain satisfied (since we increase or decrease the values of variables in both sides of constraints by the same δ). Proceed to the Main Loop. □

Note that in the Update step: a) at least one non-tight edge in $E_{\max}(X, N) \cup E_{\min}(N, X)$ becomes tight, but b) all tight edges in $E_{\max}(N, X) \cup E_{\min}(X, N)$, if any, become non-tight. Therefore, we unfortunately do not have monotonic increase of the set of tight edges. However, once a vertex obtains a tight outgoing edge, it keeps at least one such edge forever. Thus, the set of vertices possessing tight edges monotonically increases. Consequently, the sets X_0 and N_0 may only *decrease (monotonicity)*. Every increase, in the Update step, of values of vertices in N decreases the positive slacks of all edges leaving vertices in N_0 and going to X, and the positive slacks of all edges leaving vertices in X_0 and going to N (there is always at least one such edge; otherwise the algorithm terminates. (The *decrease* case 2) is analogous.) Therefore, after pseudopolynomially many steps at least one vertex in $X_0 \cup N_0$ will obtain a tight edge and will leave the set $X_0 \cup N_0$ forever. We summarize the above argument in the following

Theorem 2. *The described algorithm is pseudopolynomial, $O(|G| \cdot n \cdot W)$, where G is the size of the game graph, n the number of its vertices, and W is the largest absolute edge weight.* □

Note, retrospectively, that this algorithm is similar in spirit to the iterated potential transformation algorithm of [9] (proved exponential in [9] and pseudopolynomial in [11]). Our algorithm is based on completely different principles. Moreover, our proof and the algorithm description are considerably simpler.

8 Conclusions

The idea to describe MPGs by linear constraints and investigate the associated polytopes using linear programming methods appears natural and useful. It reveals simple algebraic properties of MPG-polytopes and allows for a new transparent LP-based algorithm for solving MPGs. In a forthcoming paper we will present further properties of MPG-polytopes and a dual algorithm, which allow for a faster convergence to a tight solution.

References

1. H. Björklund, O. Nilsson, O. Svensson, and S. Vorobyov. Controlled linear programming: Boundedness and duality. Technical Report DIMACS-2004-56, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, December 2004. `http://dimacs.rutgers.edu/TechnicalReports/`.
2. H. Björklund, O. Nilsson, O. Svensson, and S. Vorobyov. The controlled linear programming problem. Technical Report DIMACS-2004-41, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, September 2004.
3. H. Björklund, O. Svensson, and S. Vorobyov. Controlled linear programming for infinite games. Technical Report DIMACS-2005-13, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, April 2005.
4. H. Björklund, O. Svensson, and S. Vorobyov. Linear complementarity algorithms for mean payoff games. Technical Report DIMACS-2005-05, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, February 2005.
5. H. Björklund and S. Vorobyov. Combinatorial structure and randomized subexponential algorithms for infinite games. *Theoretical Computer Science*, 349(3):347–360, 2005.
6. H. Björklund and S. Vorobyov. A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. *Discrete Applied Mathematics*, 2006. Accepted, to appear. Preliminary version in MFCS'04, Springer Lecture Notes in Computer Science, vol. 3153, pp. 673-685, and DIMACS TR 2004-05.
7. R. W. Cottle, J.-S. Pang, and R. E. Stone. *The Linear Complementarity Problem.* Academic Press, 1992.
8. A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journ. of Game Theory*, 8:109–113, 1979.
9. V. A. Gurvich, A. V. Karzanov, and L. G. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 28(5):85–91, 1988.
10. K. G. Murty and F.-T. Yu. *Linear Complementarity, Linear and Nonlinear Programming.* Heldermann Verlag, Berlin, 1988.
11. N. Pisaruk. Mean cost cyclical games. *Mathematics of Operations Research*, 24(4):817–828, 1999.
12. A. Schrijver. *Theory of Linear and Integer Programming.* John Wiley and Sons, 1986.
13. A. Schrijver. *Combinatorial Optimization*, volume 1-3. Springer, 2003.
14. O. Svensson and S. Vorobyov. A subexponential algorithm for a subclass of P-matrix generalized linear complementarity problems. Technical Report DIMACS-2005-20, DIMACS: Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University, NJ, June 2005.
15. U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theor. Comput. Sci.*, 158:343–359, 1996.