Modelling Games with the help of Quantified Integer Linear Programs [⋆]

T. Ederer, U. Lorenz, T. Opfer, J. Wolf

Institute of Mathematics, Technische Universität Darmstadt, Germany

Abstract. Quantified linear programs (QLPs) are linear programs with mathematical variables being either existentially or universally quantified. The integer variant (Quantified linear integer program, QIP) is PSPACE-complete, and can be interpreted as a two-person zero-sum game. Additionally, it demonstrates remarkable flexibility in polynomial reduction, such that many interesting practical problems can be elegantly modeled as QIPs. Indeed, the PSPACE-completeness guarantees that all PSPACE-complete problems as e.g. games like Othello, Gomoku and Amazons can be described with the help of QIPs, with only moderate overhead. In this paper, we present the *Dynamic Graph Reliability* (DGR) optimization problem and the game *Gomoku* as examples.

1 Introduction

Game playing with a computer fascinates people all over the world since the beginnings of the Personal Computer, the computer for everyone. Thousands of boys and girls loved to play Atari's Pong or Namco's Pacman and many other games. The Artificial Intelligence community has picked up the scientific aspects of game playing and provided remarkable research results, especially in the area of game tree search. One example of a successful research story is coupled to the game of chess [21, 2, 11, 10, 9, 4]. Algorithmic achievements like the introduction of the Alphabeta-algorithm or the MTD(f) [1] algorithm [17] play a keyrole for the success. Currently, we observe a similar evolution in Computer-Go [20]. Here, the UCT-algorithm [2] dominates the search algorithm. Interestingly, it arose from a fruitful interplay of Theory and Practice [13, 3]. In Go, the machines increase their playing strength year by year. Some other interesting games have been completely solved [25], in others the machines dominate the human players [24].

Independently, in the 1940s, linear programming arose as a mathematical planning model and rapidly found its daily use in many industries. However, integer programming, which was introduced in 1951, became dominant far later at the beginning of the 1990s.

For traditional deterministic optimization one assumes data for a given problem to be fixed and exactly known when the decisions have to be taken. Nowadays, it is possible to solve very large mixed integer programs of practical size, but companies observe

[⋆] Research partially supported by German Research Foundation (DFG) funded SFB 805.

[1] MTD(f) or MTD(f,n): Memory-enhanced Test Driver with node n and value f)

[2] UCT: Upper Confidence bounds applied to Trees

an increasing danger of disruptions, i.e., events occur which prevent companies from acting as planned. Data are often afflicted with some kinds of uncertainties, and only estimations, maybe in form of probability distributions, are known. Examples are flight or travel times. Throughput-time, arrival times of externally produced goods, and scrap rate are subject to variations in production planning processes.

Therefore, there is a need for planning and deciding under uncertainty which, however, often pushes the complexity of traditional optimization problems, which are in P or NP, to PSPACE. These problems contain the spirit of games more then the spirit of classic optimization. An interesting result from Complexity Theory is that all NP-complete probems can be converted into each other with only moderate overhead. Analogously, all PSPACE-complete problems can be converted into each other with little effort, and the quantified versions of integer linear programs [22] cover the complexity class PSPACE. As a consequence, we can model every other problem which is PSPACE-complete or easier, with moderate (i.e. polynomial) overhead, and thus, a PSPACE complete game like Othello, Gomoku, Amazons [3] etc. is modeled via the game's rules which are encoded. It is not necessary to encode the game tree of such a game. Chess, Checkers and Go do not necessarily belong to this group of games. Their inspected extensions are even EXPTIME-hard. For the interested reader, we refer to [6, 19]. If we restrict the maximum number of allowed moves in these games by a polynomial in the input size, they are in PSPACE, and at least checkers is then PSPACE complete [7].

Suitable candidate algorithms for solving QIPs are the typical algorithms from AI like Alphabeta, UCT, Proof Number Search (PNS) [26] or MTD(f). Because of the transformation overhead from the original problem description to a QIP, there is a certain loss in solution performance. We think that this overhead might be accepable, because solving QLPs allows the fast generation of bounds to the original QIP. We are optimistic that these QLP bounds can considerably speed up the search process for the QIP, similarly as the solutions of LP-relaxations speed up the branch and bound algorithm in the context of conventional linear integer programming. However, to answer this question is a matter of ongoing research. In summery, three subtasks must be solved in order to generate huge impact on both, Game Playing and Mathematical Optimization under Uncertainty.

- Applicability must be shown. This means we have to convince that relevant problems can elegantly be modeled with the help of QIPs.
- Fast algorithms for QLPs, the relaxed versions of QIPs, must be found.
- It has to be shown that the QLP solutions can be used in order to speed up the search processes of QIPs by a huge margin. Note that this last step took about 20 years for conventional Mathematical Programming.

Thus, the idea of our research is to explore the abilities of linear programming when applied to PSPACE-complete problems, similar as it was applied to NP-complete problems in the 1990s. We could already show that QLPs have remarkable polyhedral properties [15]. Moreover, we are able to solve comparably large QLPs in reasonable time [5]. Recent research has shown that our QLP algorithm can be improved further by

[3] for an overview, cf. http://en.wikipedia.org/wiki/Game_complexity [18, 8, 12]

adopting some techniques known from the gaming community. For example, we could derive cutting planes with an interesting similarity to alpha-beta pruning, which reduce the search time by half.

2 The Problem Statement: Quantified Linear Programs

2.1 Problem statement

In the following, the definition of a *Quantified Linear Program* is stated. Let $\mathbb{Q}$ describe the set of rational numbers and $\mathbb{Z}$ the set of integer numbers.

Given: A vector of variables $x = (x_1, ..., x_n) \in \mathbb{Q}^n$, upper and lower bounds $u \in \mathbb{Z}^n$ and $l \in \mathbb{Z}^n$ with $l_i \leq x_i \leq u_i$, a matrix $A \in \mathbb{Q}^{m \times n}$, a vector $b \in \mathbb{Q}^m$ and a quantifier string $Q = (q_1, ..., q_n) \in \{\forall, \exists\}^n$, where the quantifier q_i belongs to the variable x_i, for all $1 \leq i \leq n$.

We denote a QIP/QLP as $[Q(x) : Ax \leq b]$. A maximal subset of $Q(x)$, which contains a consecutive sequence of quantifiers of the same type, is called a (variable-) *block*. A full assignment of numbers to the variables is interpreted as a game between an existential player (fixing the existentially quantified variables) and a universal player (fixing the universally quantified variables). The variables are set in consecutive order, as determined by the quantifier string. Consequently, we say that a player makes the move $x_k = z$, if he fixes the variable x_k to the value z. At each such move, the corresponding player knows the settings of $x_1, ..., x_{i-1}$ before setting x_i. If, at the end, all constraints of $Ax \leq b$ hold, the existential player wins the game. Otherwise the universal player wins.

Question: Is there an assignment for variable x_i with the knowledge, how $x_1, ..., x_{i-1}$ have been set before, such that the existential player wins the game, no matter, how the universal player acts when he has to move?
This problem occurs in two variants: a) all variables are discrete (QIP) and b) all variables are rational (QLP).

2.2 Solutions of QIPs and QLPs: Strategies and Policies

Definition 1. Strategy: *A* strategy *for the existential player S is a tuple* $(V_x \dot\cup V_y, E, L)$, *that is, a labeled tree of depth* n *with vertices* V *and edges* E *(*$|V|$ *and* $|E|$ *being their respective sizes), where* V_x *and* V_y *are two disjoint sets of nodes, and* $L \in \mathbb{Q}^{|E|}$ *is a set of edge-labels. Nodes from* V_x *are called* existential nodes, *nodes from* V_y *are called* universal nodes. *Each tree level* i *consists either of only existential nodes or of only universal nodes, depending on the quantifier* q_i *of the variable* x_i*. Each edge of the set* E*, leaving a tree-node of level* i*, represents an assignment for the variable* x_i*.* $l_i \in L$ *describes the value of variable* x_i *on edge* $e_i \in E$*. Existential nodes have exactly one successor, universal nodes have as many successors as the universal player has choices at that node. In case of QLPs, it suffices to deal with two successors, induced by the upper and the lower integer bounds of a universal variable, below each universal node*

[15]. A strategy is called a winning strategy*, if all paths from the root to a leaf represent a vector x such that $Ax \leq b$.*

Definition 2. Policy: *A* policy *is an algorithm that fixes a variable x_i, being the i^{th} component of the vector x, with the knowledge, how $x_1, ..., x_{i-1}$ have been set before.*

A three-dimensional example of a QIP/QLP is given below:

$$\forall x_1 \in [-1, 0]\ \forall x_2 \in [0, 1]\ \exists x_3 \in [-2, 2] : \quad \begin{pmatrix} 10 & -4 & 2 \\ 10 & 4 & -2 \\ -10 & 4 & 1 \\ -10 & -4 & -1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 4 \\ 12 \\ 8 \end{pmatrix}$$

If we restrict the variables in the example to the integer bounds of their domains, we observe a winning strategy for the existential player as shown in Figure 1. In this example, a + in a tree leaf means that the existential player wins when this leaf is reached. A - marks a win for the universal player. Numbers at the edges mark the choices for variables. If the universal player moves to -1 and 0 (i.e., he sets $x_1 = -1$ and $x_2 = 0$) the existential player has to move to 2. If the universal player moves to -1 and 1, the exis-

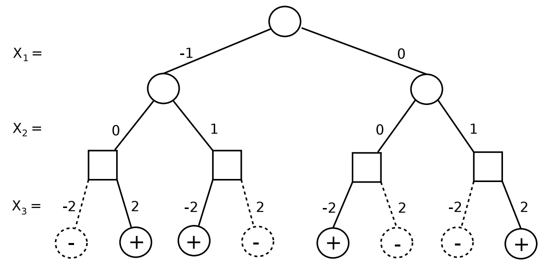


Fig. 1. The winning strategy (solid) for the integer QLP example above

tential player must set the variable $x_3 = -2$ etc. We see that the existential player has to react carefully to former moves of the universal player. If we now relax the variables, allowing non-integral values for the corresponding domains, the resulting solution space of the corresponding QLP becomes polyhedral. Moreover, the solution of the resulting problem when $x_1 = -1$ is a line segment: B (cf. Fig 2). On the left side of Figure 2, we can still see the corresponding partial strategy of the integer example in Figure 1. The strategy consists of the two end-points of B. On the right side of Figure 2, the same convex hull of the solution space is shown from another perspective. We observe that the existential player has more freedom in the choice of x_3, when the universal player sets $x_1 = 0$. If $x_1 = 0$, the solution space of the rest-problem will just be the facet C.

Remark: If the integrality constraints of a QIP are relaxed, this will result in a QLP. One may distinguish between QLPs where only the variables of the existential player

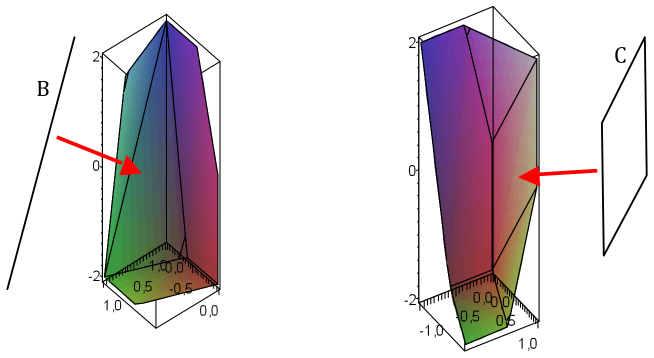


Fig. 2. A visualization of the 3-d solution space of the example above

are relaxed, and QLPs where all variables are relaxed. However, these two variants are equivalent to each other, as in [15] it was shown that

- Let $y_1, .., y_m$ be the universal variables of a given QLP. Let $l_1, ..., l_m$ be lower bound restrictions to the y-variables and let $u_1, ..., u_m$ be the corresponding upper bounds. The existential player has a winning strategy against the universal player who takes his choices $y_i \in \{l_i, u_i\}, i \in 1...m$ if and only if the existential player has a winning policy against the universal player who takes his choices from the corresponding rational intervals, i.e. $y_i \in [l_i, u_i], i \in 1...m$.
- Whether or not there is a winning strategy for the existential player can be determined with polynomially many bits, as long as the number of quantifier changes is constant, independently of the number of variables.

3 Modelling with QIPs

Two different games are modeled with the help of QIPs in this section. The first one is a simple graph game, where a person has to travel on a graph from a given source node to a desired target node. While he is traveling, an (evil) opponent erases some edges and thus destroys the graph. However, also the opponent must follow certain rules. The second game, which is inspected, is the well known *Five in a Row* game. Both are PSPACE-complete, and can therefore not be modeled on small space by propositional logic or by mixed integer linear programming (except PSPACE=NP).

3.1 A two-person zero-sum graph game

In order to demonstrate the modelling power of QIPs, we firstly present an example for the so called worst-case Dynamic Graph Reliability problem. It is closely related to the QSAT-problem and to the Dynamic Graph Reliability problem (DGR) [16]. A variant of this game was re-invented by van Benthem [23] and analyzed by Löding and Rhode [14].

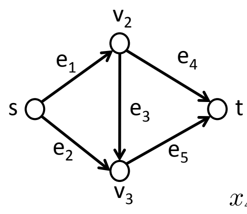
A worst-case DGR (wcDGR) is defined as follows.

Given: A directed acyclic graph $G = (V, E)$ with two special nodes s and t. Moreover, we have defined a mapping $f : (V \times E) \to \{0, 1\}$ with $f(v, e) = 1$ if and only if an opponent is allowed to erase edge e when we arrive at node v. Moreover, the opponent has to follow some rules as well. For some edges, the opponent is not allowed to erase more than one of two specific edges. In other words, there is another mapping $g : E \times E \to \{0, 1\}$ with $g(e_1, e_2) = 1$ if and only if it is allowed to erase both e_1 and e_2.

Question: Is there a strategy, which allows the existential player to reach the target node t from start node s, no matter, how the opponent acts?

Starting point of the wcDGR is a directed acyclic graph (DAG). An example is shown on the left side of Figure 3. There, we assume that an opponent may erase at most one of the edges e_4 and e_5. He can make them fail when we arrive at node v_2 or at node v_3. Anyway, never both edges are allowed to fail, and no other edges can fail. The optimization problem is to firstly make a choice whether to travel via edge e_1 or e_2. Then, the opponent erases none or one of the edges e_4 and e_5. Thereafter, we choose a remaining path to target t, if one exists. If we move from node v_2 to node v_3, our opponent is again allowed to make one of the two edges e_4 or e_5 fail.

Let us introduce variables $x_1, ..., x_5$ for edge-choices. $x_i = 1$ means e_i is chosen for traveling. The first block of constraints encodes the flow constraints of the classic shortest-path problem on graphs. The constraints are applied to the x-variables (Fig. 3, right, (1)). $y_{2,4}, y_{3,4}, y_{2,5}, y_{3,5}$ determine whether the opponent makes the edges e_4 or e_5 fail when we reach the nodes v_2 or v_3, i.e. $y_{i,j} = 1$ means that there is a failure on



$$\exists x_1, x_2 \forall y_{2,4}, y_{2,5} \exists x_3 \forall y_{3,4}, y_{3,5} \exists x_4, x_5, x_\Delta \text{ (all binary) :}$$

$$\left.\begin{array}{r} x_1 = x_4 + x_3 \\ x_2 + x_3 = x_5 \\ x_1 + x_2 = 1 \\ x_4 + x_5 = 1 \end{array}\right\} \text{flow constraints (1)}$$

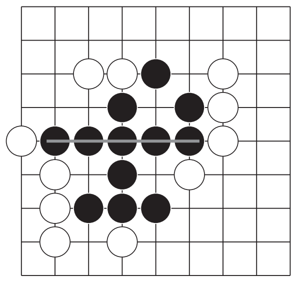
$$\left.\begin{array}{l} x_4 \leq (1 - y_{2,4}) + (1 - x_1) + x_\Delta \\ x_4 \leq (1 - y_{3,4}) + (1 - x_2 - x_3) + x_\Delta \\ x_5 \leq (1 - y_{2,5}) + (1 - x_1) + x_\Delta \\ x_5 \leq (1 - y_{3,5}) + (1 - x_2 - x_3) + x_\Delta \end{array}\right\} \begin{array}{l}\text{failure constraints} \\ \text{for the existential} \\ \text{player (2)}\end{array}$$

$$2x_\Delta \leq y_{3,4} + y_{3,5} + y_{2,4} + y_{2,5} \} \begin{array}{l}\text{(3) critical failure} \\ \text{constraint for the} \\ \text{universal player.}\end{array}$$

Fig. 3. graph of wcDGR example and QIP description edge e_j. The second block (2) couples the decision variables x_i to the $y_{j,k}$-variables of the opponent. E.g. $x_4 \leq (1 - y_{2,4}) + (1 - x_1) + x_\Delta$ means that the existential player will have to set x_4 to zero and will not be allowed to use edge e_4 if the existential player first moves via edge e_1 and then the universal player sets $y_{2,4} = 1$. Strictly seen, we have to test whether the existential player has moved via node v_2. However, a directed graph can always be pre-manipulated such that all nodes of the original graph can be entered via one specific incoming edge. Of course, for this purpose, additional nodes and edges must be added, in general. The variable x_Δ is used to ensure that also the universal player follows his rules. The last constraint (3) $2x_\Delta \leq y_{3,4}+y_{3,5}+y_{2,4}+y_{2,5}$ expresses that the universal player is constrained by $y_{3,4} + y_{3,5} + y_{2,4} + y_{2,5} \leq 1$. If he breaks this rule, the existential player can set $x_\Delta = 1$ and the constraints of the second block are trivialized. The existential player can then trivially win the game. Last but not least, we can express the problem as shown on the right side of Fig. 3, with the given quantifier-prefix, because the graph of a wcDGR is a DAG and therefore a partial order (in time) of the nodes can be computed.

3.2 Gomoku

Five in a Row or *Gomoku* is a two-person strategy game played with *Go* pieces on a *Go* board. Both players (black and white, black begins) alternately place stones, until the first player gets an row of five horizontally, vertically or diagonally connected stones. A once placed stone cannot be moved or removed from the board. With this standard set of rules, it is known that black always wins on some board sizes (e.g. 15×15 [1]), but the problem is open for arbitrary $n \times n$ boards.



Parameters
As we showed in the preceeding paragraph, we can model that also the universal player has to follow some rules with auxiliary existential variables. We simplify the description of Gomoku by not considering this detail. Instead, we only present the rules for the universal player. Let T denote the set of all moves (there are up to n^2 moves until the board is full) and N^2 the set of all board coordinates. Let H^2 be the set of coordinates of a reduced board (used to detect connected rows of five stones) and C a counting set. The decision parameters δ_t specify, which player places a stone in move t.

$$T := \{1, \dots, n^2\}, \quad N := \{1, \dots, n\},$$
$$H := \{1, \dots, n-4\}, \quad C := \{0, \dots, 4\},$$
$$\delta_t^b := t \bmod 2, \quad \delta_t^w := 1 - (t \bmod 2)$$

Variables
Let $\mathbb{B} = \{0, 1\}$ denote the binary set. For each move $t \in T$ and each field with coordinates $(i, j) \in N^2$, let there be two binary variables $x_{t,i,j}^b$ indicating a black stone

and $x^w_{t,i,j}$ indicating a white stone. Using the set $\{b, w\}$ as upper index implies that a statement is equally valid for both black and white stones.

For each move $t \in T$ and coordinate $(k, l) \in N \times H$, let $h^{\{b,w\}}_{t,k,l}$ be an indicator variable denoting the existence of a horizontally connected black or respectively white row of five stones beginning on this coordinate. The range of column indices is slightly smaller than N, because each such connected row cannot start near the right board edge. Analogously, let there be vertical indicator variables $v^{\{b,w\}}_{t,k,l}$ for each move $t \in T$ and coordinate $(k, l) \in H \times N$ and diagonal indicator variables $d^{\{b,w\}}_{t,k,l}$ for each move $t \in T$ and coordinate $(k, l) \in H^2$.

Using the connected row indicators, winning criteria can be expressed. Let $s^{\{b,w\}}_t$ be a monotonously increasing indicator function (*step function*), which raises the first time a player has any connected row of five. Further, let $p^{\{b,w\}}_t$ be an indicator function with norm one (*peak function*), which peaks the first time a player has any connected row of five and is false otherwise. The event of winning the game can be expressed by indicator variables $e^{\{b,w\}}_t$, which is true if and only if a player gets any connected row of five for the first time and his opponent did not get any connected row of five before. The retaliation indicator $r^{\{b,w\}}$ indicates, if a player does not win at all.

$$x^{\{b,w\}}_{t,i,j} \in \mathbb{B}^{|T\times N^2|}$$

$$h^{\{b,w\}}_{t,k,l} \in \mathbb{B}^{|T\times N\times H|},\ v^{\{b,w\}}_{t,k,l} \in \mathbb{B}^{|T\times H\times N|},\ d^{\{b,w\}}_{t,k,l}, u^{\{b,w\}}_{t,k,l} \in \mathbb{B}^{|T\times H^2|}$$

$$s^{\{b,w\}}_t \in \mathbb{B}^{|T|},\ p^{\{b,w\}}_t \in \mathbb{B}^{|T|},\ e^{\{b,w\}}_t \in \mathbb{B}^{|T|},\ r^{\{b,w\}} \in \mathbb{B}$$

To express the quantifier string in a compact form, we denote the following abbreviations:

$$\begin{aligned} & A_0 \coloneqq \left\{\forall\,(i,j) \in N^2 : x^{\{b,w\}}_{0,i,j},\ s^{\{b,w\}}_0\right\} \\ \forall t \in T: \quad & A_t \coloneqq \Big\{\forall\,(i,j) \in N^2 : x^{\{b,w\}}_{t,i,j},\ \forall\,(k,l) \in N \times H : h^{\{b,w\}}_{t,k,l}, \\ & \qquad \forall\,(k,l) \in H \times N : v^{\{b,w\}}_{t,k,l},\ \forall\,(k,l) \in H^2 : d^{\{b,w\}}_{t,k,l}, \\ & \qquad s^{\{b,w\}}_t,\ p^{\{b,w\}}_t,\ e^{\{b,w\}}_t\Big\} \end{aligned}$$

Optimization Model
The solution of a quantified optimization model is the information, if the player (in this case black) can win against his opponent. If he can win, the first move leading to the victory is provided. (If one wishes to play a full *Gomoku* game using a quantified model, one therefore has to solve the model after each move of the opponent.)

We choose to extend the basic model with an objective function to rate black's strategy. If black can win at all, the optimal objective value is the first possible move in which black can win. If black cannot win, but he can force a draw, the optimal objective value is $(n^2 + 1)$. If black definitely loses, the optimal objective value minus $(n^2 + 1)$ depicts the longest possible delay black can achieve until he is beaten. To extract the basic information if black can win, the objective value can be compared to the remis value (n^2+1).

minimize

$$\sum_{t\in T} t\cdot e_t^b + (2n^2+1)\cdot r^b - \sum_{t\in T} t\cdot e_t^w - n^2\cdot r^w$$

such that

$$\exists\, A_0\ \ \exists\, A_1\ \forall\, A_2\ \exists\, A_3\ \forall\, A_4\ \ldots\ \exists\forall\exists\forall\ \ldots\ A_{n^2}\ \ \exists\, r^{\{b,w\}}$$

$$\forall\,(i,j)\in N^2:\ x_{0,i,j}^{\{b,w\}}=0,\quad s_0^{\{b,w\}}=0 \tag{1}$$

(initial state: All intersections are free. both players have not won yet.)

$$\forall\, t\in T,\ \forall\,(i,j)\in N^2:\quad x_{t,i,j}^b + x_{t,i,j}^w \le 1 \tag{2}$$

(occupation: On every intersection no more than one stone can be placed.)

$$\forall\, t\in T,\ \forall\,(i,j)\in N^2:\quad x_{t,i,j}^{\{b,w\}} \ge x_{t-1,i,j}^{\{b,w\}} \tag{3}$$

(causality: Every once placed stone remains placed.)

$$\forall\, t\in T:\quad \sum_{(i,j)\in N^2} x_{t,i,j}^{\{b,w\}} = \sum_{(i,j)\in N^2} x_{t-1,i,j}^{\{b,w\}} + \delta_t^{\{b,w\}} \tag{4}$$

(alternation: In every odd move black places exactly one stone. In every even move white places exactly one stone. In both cases the opposite player remains idle.)

$$\forall\, t\in T,\ \forall\,(k,l)\in N\times H,\ \forall\, c\in C:\ h_{t,k,l}^{\{b,w\}} \le x_{t,k,l+c}^{\{b,w\}} \tag{5a}$$

$$\forall\, t\in T,\ \forall\,(k,l)\in N\times H\qquad :\ h_{t,k,l}^{\{b,w\}} \ge \sum_{c\in C} x_{t,k,l+c}^{\{b,w\}} - 4 \tag{5b}$$

(horizontal rows: a player got a horizontal row beginning on coordinate (k,l), if and only if the specified and the next four intersections to the right contain stones of the same color.)

$$\forall\, t\in T,\ \forall\,(k,l)\in H\times N,\ \forall\, c\in C:\ v_{t,k,l}^{\{b,w\}} \le x_{t,k+c,l}^{\{b,w\}} \tag{6a}$$

$$\forall\, t\in T,\ \forall\,(k,l)\in H\times N\qquad :\ v_{t,k,l}^{\{b,w\}} \ge \sum_{c\in C} x_{t,k+c,l}^{\{b,w\}} - 4 \tag{6b}$$

(vertical rows: a player got a vertical row beginning on coordinate (k,l), if and only if the specified and the next four intersections to the bottom contain stones of the same color.)

$$\forall\, t \in T,\ \forall\,(k,l) \in H^2,\ \forall\, c \in C: \quad d^{\{b,w\}}_{t,k,l} \leq x^{\{b,w\}}_{t,k+c,l+c} \tag{7a}$$
$$\forall\, t \in T,\ \forall\,(k,l) \in H^2 \quad : \quad d^{\{b,w\}}_{t,k,l} \geq \sum_{c \in C} x^{\{b,w\}}_{t,k+c,l+c} - 4 \tag{7b}$$

(downward diagonal rows: a player got a downward diagonal row beginning on coordinate (k, l), if and only if the specified and the next four intersections to the bottom-right contain stones of the same color.)

$$\forall\, t \in T,\ \forall\,(k,l) \in H^2,\ \forall\, c \in C: \quad u^{\{b,w\}}_{t,k,l} \leq x^{\{b,w\}}_{t,k+c,l+4-c} \tag{8a}$$
$$\forall\, t \in T,\ \forall\,(k,l) \in H^2 \quad : \quad u^{\{b,w\}}_{t,k,l} \geq \sum_{c \in C} x^{\{b,w\}}_{t,k+c,l+4-c} - 4 \tag{8b}$$

(upward diagonal rows: a player got an upward diagonal row beginning on coordinate $(k, l + 4)$, if and only if this and the next four intersections to the top-right contain stones of the same color. Heed that the upward diagonal row with index (k, l) does *not* contain a stone on intersection (k, l).)

$$\forall\, t \in T \quad : \quad s^{\{b,w\}}_{t} \geq s^{\{b,w\}}_{t-1} \tag{9a}$$
$$\forall\, t \in T,\ \forall\,(k,l) \in N \times H: \quad s^{\{b,w\}}_{t} \geq h^{\{b,w\}}_{t,k,l} \tag{9b}$$
$$\forall\, t \in T,\ \forall\,(k,l) \in H \times N: \quad s^{\{b,w\}}_{t} \geq v^{\{b,w\}}_{t,k,l} \tag{9c}$$
$$\forall\, t \in T,\ \forall\,(k,l) \in H^2: \quad s^{\{b,w\}}_{t} \geq d^{\{b,w\}}_{t,k,l} \tag{9d}$$
$$\forall\, t \in T,\ \forall\,(k,l) \in H^2: \quad s^{\{b,w\}}_{t} \geq u^{\{b,w\}}_{t,k,l} \tag{9e}$$

$$\forall\, t \in T: \quad s^{\{b,w\}}_{t} \leq s^{\{b,w\}}_{t-1} + \sum_{(k,l) \in N \times H} h^{\{b,w\}}_{t,k,l} + \sum_{(k,l) \in H \times N} v^{\{b,w\}}_{t,k,l} + \sum_{(k,l) \in H^2} d^{\{b,w\}}_{t,k,l} + \sum_{(k,l) \in H^2} u^{\{b,w\}}_{t,k,l} \tag{9f}$$

(row history: The monotonously increasing indicator function s raises in the unique move t, when the player gets a row for the first time.)

$$\forall\, t \in T: \quad p^{\{b,w\}}_{t} = s^{\{b,w\}}_{t} - s^{\{b,w\}}_{t-1} \tag{10}$$

(critical move: The indicator function p with norm one peaks in the unique move t, when the player gets a row for the first time.)

$$\forall\, t \in T: \quad e_t^b \leq p_t^b \tag{11a}$$

$$\forall\, t \in T: \quad e_t^b \leq (1 - s_t^w) \tag{11b}$$

$$\forall\, t \in T: \quad e_t^b \geq p_t^b + (1 - s_t^w) - 1 \tag{11c}$$

$$\forall\, t \in T: \quad e_t^w \leq p_t^w \tag{11d}$$

$$\forall\, t \in T: \quad e_t^w \leq (1 - s_t^b) \tag{11e}$$

$$\forall\, t \in T: \quad e_t^w \geq p_t^w + (1 - s_t^b) - 1 \tag{11f}$$

(victory: A player wins in the unique move t, if and only if he gets his first row in move t and his opponent has not done so before move t. Heed the switched *b/w* indices.)

$$r^{\{b,w\}} + \sum_{t \in T} e_t^{\{b,w\}} = 1 \tag{12}$$

(retaliation: If a player does not win at all, his retaliation indicator is activated. This information is used to weight ties and defeats in the objective function.)

After all, we see that Gomoku could elegantly be described with the help of a QIP. An obvious drawback is that the number of constraints has grown by a factor of n^2. However, it is neither the case that a short description necessarily leads to faster solutions, nor do we claim that our description is the shortest possible one.

4 Conclusion

Quantified Linear Integer Programs form quite a powerful and elegant modelling tool. They have structural properties of Linear Programs on the one hand, but on the other hand, they describe the complexity class PSPACE and therefore are natural candidates for the modeling of games like graph games, Go-Moku, Sokoban and many more. We guess that this modeling technique has the potential to bridge the gap between Mathematical Optimization and Game Playing in the Artificial Intelligence. In this paper, we could show how to model some games with the help of QIPs. Moreover, we reported about progress solving QLPs. Future work will show whether QLPs can be used to speed up search times for QIPs in a similar way as MIPs are speeded up with the help of LP-relaxation. If QLPs are able to play a similar role for QIPs as LPs play for IPs, we can hope for a performance jump, at least for exact solving of PSPACE-complete games.

References

1. L.V. Allis. Searching for solutions in games and artificial intelligence. In *Ph.D. thesis*, 1994.
2. J.H. Condon and K. Thompson. Belle chess hardware. *Advances in Computer Chess III, M.R.B. Clarke (Editor), Pergamon Press*, pages 44–54, 1982.
3. R. Coulom. Efficient selectivity and backup operators in monte-carlo tree search. *Proc. of Computers and Games 2006*, pages 72–83, 2006.

4. C. Donninger and U. Lorenz. The chess monster hydra. *Proc. of International Conference ob Field-Programmable Logic and Applications (FPL)*, pages 927–932, 2004. LNCS 3203, Antwerp - Begium.
5. Thorsten Ederer, Ulf Lorenz, Alexander Martin, and Jan Wolf. Quantified linear programs: A computational study. In *accepted for European Symposium on Algorithms (ESA)*. Springer, 2011.
6. A. Fraenkel and D. Lichtenstein. Computing a perfect strategy for n$\times$n chess requires time exponential in n. *J. Comb. Th. A*, 31:199–214, 1981.
7. A. S. Fraenkel, M. R. Garey, D. S. Johnson, T. Schaefer, and Y. Yesha. The complexity of checkers on an n $\times$ n board. In *19th Annual Symposium on Foundations of Computer Science (FOCS 1978)*, pages 55–64, 1978.
8. R.A. Hearn. Amazons is pspace-complete. Technical Report cs.CC/0502013, Feb 2005.
9. F-H. Hsu. Ibm's deep blue chess grandmaster chips. *IEEE Micro*, 18(2):70–80, 1999.
10. F-H. Hsu, T.S. Anantharaman, M.S. Campbell, and No. Deep thought. *Computers, Chess, and Cognition*, pages 55–78, 1990.
11. R.M. Hyatt, B.E. Gower, and Nelson H.L. Cray blitz. *Advances in Computer Chess IV, D.F. Beal (Editor), Pergamon Press*, pages 8–18, 1985.
12. S. Iwata and T. Kasai. The othello game on an n*n board is pspace-complete. *Theoretical Computer Science*, 123:329–340, 1994.
13. L. Kocsis and C. Szepesvari. Bandit based monte-carlo planning. *Proc. of ECML 2006, Springer LNAI 4212, Berlin*, pages 282–293, 2006.
14. C. Loeding and P. Rohde. Solving the sabotage game is pspace-hard. In *28th International Symposium on Mathematical Foundations of Computer Science, MFCS*, LNCS 2747, pages 531–540, 2003.
15. U. Lorenz, A. Martin, and J. Wolf. Polyhedral and algorithmic properties of quantified linear programs. *Annual European Symposium on Algorithms*, pages 512–523, 2010.
16. C.H. Papadimitriou. Games against nature. *J. of Comp. and Sys. Sc.*, pages 288–301, 1985.
17. Aske Plaat, Jonathan Schaeffer, Wim Pijls, and Arie De Bruin. Best-first fixed-depth game-tree search in practice. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1*, pages 273–279, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.
18. S. Reisch. Gobang ist pspace-vollstandig (gomoku is pspace-complete). *Acta Informatica*, 13: 5966, 1999.
19. J. M. Robson. The complexity of go. In *Proceedings of IFIP Congress*, pages 413–417, 1983.
20. D. Silver. *Reinforcement Learning and Simulation-Based Search in Computer Go*. PhD thesis, University of Alberta, 2009.
21. D.J. Slate and L.R. Atkin. Chess 4.5 - the northwestern university chess program. *Chess Skill in Man and Machine, P.W. Frey (Editor), Springer Verlag*, pages 82–118, 1977.
22. K. Subramani. Analyzing selected quantified integer programs. *Springer, LNAI 3097*, pages 342–356, 2004.
23. J. van Benthem. An essay on sabotage and obstruction. In *Festschrift in Honour of Prof. Joerg Siekmann*, LNAI, 2002.
24. H.J. van den Herik, J. Nunn, and D. Levy. Adams outclassed by hydra. *ICGA Journal*, 28(2):107–110, 2005.
25. H.J. van den Herik, J.W.H.M. Uiterwijk, and J. van Rijswijk. Games solved: Now and in the future. *Artificial Intelligence*, 134:277–312, 2002.
26. Mark H.M. Winands, Jos W.H.M. Uiterwijk, and Jaap van den Herik. Pds-pn: A new proof-number search algorithm. In *Computers and Games (CG)*, pages 61–74, 2002.