Computing Game Values for Crash Games

Thomas Gawlitza and Helmut Seidl

TU München, Institut für Informatik, I2
85748 München, Germany
{gawlitza, seidl}@in.tum.de

Abstract. We consider crash games which are a generalization of parity games in which the play value of a play is an integer, $-\infty$ or ∞. In particular, the play value of a finite play is given as the sum of the payoffs of the moves of the play. Accordingly, one player aims at maximizing the play value whereas the other player aims at minimizing this value. We show that the game value of such a crash game at position v, i.e., the least upper bounds to the minimal play value that can be enforced by the maximum player in a play starting at v, can be characterized by a hierarchical system of simple integer equations. Moreover, we present a practical algorithm for solving such systems. The run-time of our algorithm (w.r.t. the uniform cost measure) is independent of the sizes of occurring numbers. Our method is based on a strategy improvement algorithm. The efficiency of our algorithm is comparable to the efficiency of the discrete strategy improvement algorithm developed by Vöge and Jurdzinski for the simpler Boolean case of parity games [19].

1 Introduction

Crash games are a generalization of parity games where game positions have non-negative ranks and additionally, each possible move of a player comes with a payoff in $\mathbb{Z}$. A play is played by two opponents, the $\vee$-player and the $\wedge$-player. The $\vee$-player wants to maximize the play value while the $\wedge$-player wants to minimize it. The play value of a finite play is determined as the sum of payoffs of moves chosen in the play. The play value of an infinite play, on the other hand, is determined similarly as for parity games: If the least rank of an infinitely often visited position is *odd*, then the $\vee$-player wins, i.e., the play value is ∞. Accordingly, if the least rank of an infinitely often visited position is *even*, then the $\wedge$-player wins, i.e., the play value is $-\infty$.

Thus, crash games are *payoff* games. The notable difference to *mean-payoff* games, for instance, is the fact that the goal for crash games is not to maximize (resp. minimize) the *mean* payoff during a play but the *total* payoff. Similar to *mean-payoff parity games* [3], play values are not only determined by the payoff function but also by a rank function. Also similar to *mean-payoff parity games*, winning strategies are no longer necessarily *positional* (also called *memoryless*) [9, 10]. Instead already for quite simple crash games, unbounded memory is required. Another class of games related to crash games are the *longest shortest paths games* from [14]. In contrast to our games, the max player in longest shortest path games is bound to use a *positional* strategy which is not the case in our setting. Also, longest shortest path games do not consider ranks for positions in the game graph.

In this paper we present basic techniques for dealing with crash games. In particular, we show that computing the game values of a crash game can be reduced to solving hierarchical systems of equations over the complete lattice $\mathcal{Z} = \mathbb{Z} \cup \{-\infty, \infty\}$. The occurring equations are *simple*, i.e., they only use the operations maximum, minimum as well as addition restricted to at most one non-constant argument. Since the lattice has infinite strictly ascending and descending chains, extra insights are necessary to solve hierarchical systems of such equations. Our main technical contribution therefore is to provide a fast practical algorithm for solving these systems.

In hierarchical systems least and greatest fixpoint variables alternate. Such systems naturally occur when model-checking finite-state systems w.r.t. temporal logics formulas (see, e.g., [1]). While classically, only two-valued logics are considered, more general complete lattices have attracted attention recently [4, 18]. The approaches in [4, 18] are restricted to finite lattices. In [17] the complete lattice of the non-negative integers extended with ∞ is considered. This lattice is of infinite height and hierarchical systems over this lattice allow to analyze quantitative aspects of the behavior of finite-state systems [17]. Opposed to that paper, we here allow also negative integers. Our algorithm for solving hierarchical systems is based on *strategy improvement*. Strategy improvement has been introduced by Howard for solving stochastic control problems [12, 16]. For the two-valued logic case, *strategy improvement algorithms* has been suggested for model-checking for the modal μ-calculus as well as for computing game values of parity games [11, 15, 19].

Our strategy improvement algorithm works directly on the hierarchical system. Thereby a strategy is a function which selects for every expression $e_1 \vee e_2$ ("$\vee$" denotes the maximum-operator) one of the subexpressions e_1, e_2. Thus, a strategy describes which side of a maximum-expression should be used and characterizes a hierarchical system in which maximum-operators do not occur any more. In general, strategy improvement algorithms try to find optimal strategies. Therefore they compute a valuation for a given strategy which, if the strategy is not yet optimal, gives hints on how to improve the strategy locally. In our case the valuation is given as the canonical solution of the system which is described by the strategy.

We have not found a technique to apply this idea to general integer systems directly. Instead, we first consider the case of integer systems where all solutions are guaranteed to be finite. In this case, we can *instrument* the underlying lattice in such a way that the resulting system has exactly one solution — from which the canonical solution of the original system can be read off. The lattice obtained by our instrumentation is closely related to the progress measures proposed by Jurdzinski for computing the winning positions in parity games [13]. Our technique is more general as it also allows to deal with integers instead of booleans. The interesting idea of Jurdzinski (cast in our terminology) is to instrument the Boolean lattice just to replace all greatest fixpoints by unique fixpoints. By this, computing canonical solutions over the Boolean lattice is reduced to computing *least* solutions over the instrumented lattice. A similar idea can also be applied in the integer case — given that the canonical solution is finite. The resulting algorithm, however, will not be *uniform*, i.e., its run-time (w.r.t. the uniform cost measure where, e.g., arithmetic operations are counted for $\mathcal{O}(1)$) may depend on the sizes of occurring numbers. Instead, our instrumentation allows us to construct a uni-

form algorithm for computing canonical solutions (given that they are finite) through a generalization of the strategy iteration techniques in [5, 8].

Using any method for computing finite canonical solutions as a subroutine, we solve the unrestricted case in two stages. First, we design an algorithm which can deal with positive infinities in the equation system. Repeatedly applying this algorithm then allows us to deal with systems whose canonical solutions may both contain $-\infty$ and ∞.

The rest of the paper is organized as follows. In section 2 we introduce crash games and give simple examples. In section 3, we introduce hierarchical systems of simple equations over $\mathcal{Z}$, which we use as a tool for solving crash games. The corresponding reduction will be discussed in section 4. In section 5, we present our key technical idea for computing canonical solutions. There, we are first restricted to hierarchical systems with a finite canonical solution. In section 6, we apply the developed technique to derive a method for computing canonical solutions without this restriction. We conclude with section 7.

2 Crash Games

In this section we introduce crash games. Such games are played by a $\vee$-player and a $\wedge$-player. They model the situation where two opponents want to maximize (resp. minimize) their total sums of investments and rewards. With each play we therefore associate a play value form the complete lattice $\mathcal{Z} = \mathbb{Z} \cup \{-\infty, \infty\}$, where $-\infty$ and ∞ denote the least and the greatest element, respectively.

The crash game G itself is given by a finite labeled graph whose nodes are equipped with non-negative ranks and whose edges carry payoffs in $\mathbb{Z}$. We assume that every node has at least one out-going edge — besides a distinguished sink node $\mathbf{0}$ indicating the end of finite plays. Each non-sink node either is a $\vee$- or a $\wedge$-node. At a $\vee$-node, the $\vee$-player may choose one of the out-going edges; likewise at a $\wedge$-node, the $\wedge$-player has a choice. The value of a play reaching 0 is the sum of the payoffs of edges chosen during the play. For infinite plays, the play values $-\infty$ or ∞ are determined similarly to the play values of plays in a parity game. Formally, we define a *crash game* as a tuple $G = (V_\vee, V_\wedge, E, c, r)$ where

1. $V_\vee$ and $V_\wedge$ are disjoint finite sets of *positions* that belong to the $\vee$-player and the $\wedge$-player, respectively. The set of all *positions* is the set $V = V_\vee \cup V_\wedge \cup \{\mathbf{0}\}$ where $\mathbf{0} \notin V_\vee \cup V_\wedge$.
2. $E \subseteq V^2$ is the set of *moves* where $\{v\}E \neq \emptyset$ for all $v \in V_\vee \cup V_\wedge$ and $\{\mathbf{0}\}E = \emptyset$. [1] This means that every position besides the position $\mathbf{0}$ must have a successor. The position $\mathbf{0}$ must not have a successor.
3. $c : E \to \mathbb{Z}$ is the *payoff function* which associates a *payoff* to each move.
4. $r : V_\vee \cup V_\wedge \to \mathbb{N}$ is the *rank function* which associates a natural number to each position from $V_\vee \cup V_\wedge$.

A *finite play* over G is a finite word $\pi = v_1 \cdots v_k$ with $v_k = \mathbf{0}$ and $(v_i, v_{i+1}) \in E$ for all $i = 1, \ldots, k-1$. The *play value* $val_G(\pi)$ of the finite play π is the

[1] For $E \subseteq V^2$ and $V' \subseteq V$, $V'E$ denotes the set $\{v_2 \in V \mid \exists v_1 \in V' \text{ such that } (v_1, v_2) \in E\}$.

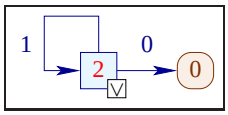
sum $\sum_{i=1}^{k-1} c((v_i, v_{i+1}))$. Accordingly, an *infinite* play over G is an infinite word $\pi = v_1 v_2 \cdots$ with $(v_i, v_{i+1}) \in E$ for all $i \in \mathbb{N}$. Assume that m denotes the natural number $\min\{r(v) \in V_\vee \cup V_\wedge \mid v \text{ occurs infinitely often in } \pi\}$. The *play value* $val_G(\pi)$ then is ∞ if m is odd and $-\infty$ otherwise. By $Play_G$ we denote the set of all plays over G and by $Play_G(v)$ the set of all plays starting at $v \in V$, i.e., $Play_G(v) = Play_G \cap \{v\} \cdot (V^\omega \cup V^*)$.

For a finite play $\pi = v_1 \cdots v_k$ (resp. infinite play $\pi = v_1 v_2 \cdots$) the set of *prefixes of* π is the set $\{v_1 \cdots v_i \mid i = 0, \ldots, k\}$ (resp. $\{v_1 \cdots v_i \mid i \in \mathbb{N}_0\}$) which we denote by $Prefix(\pi)$. The set of all prefixes of plays over G ending in a $\vee$-position (resp. $\wedge$-position) is denoted by $Prefix_\vee(G)$ (resp. $Prefix_\wedge(G)$). For a play prefix $\pi = v_1 \cdots v_k$ we write $c(\pi)$ for the sum $\sum_{i=1}^{k-1} c((v_i, v_{i+1}))$. A $\vee$-strategy (resp. $\wedge$-strategy) is a function $f : Prefix_\vee(G) \to V$ (resp. $f : Prefix_\wedge(G) \to V$) where, for every play prefix π, $\pi \cdot f(\pi)$ is again a play prefix. A $\vee$-strategy (resp. $\wedge$-strategy) f is *positional* iff $f(\pi v) = f(\pi' v)$ for all play prefixes $\pi v, \pi' v$ ending in the same $\vee$-position v (resp. $\wedge$-position v). We write $F_\vee(G)$ for the set of all $\vee$-strategies and $F_\wedge(G)$ for the set of all $\wedge$-strategies. The play π is *consistent* with the $\vee$-strategy f (resp. $\wedge$-strategy f) iff for every finite prefix $\pi' v$ of π, $f(\pi') = v$ whenever $\pi' \in Prefix_\vee(G)$ (resp. $\pi' \in Prefix_\wedge(G)$). For a set P of plays, we write $P|_f$ for the set of plays from P that are consistent with f. For a position v, we define the *game value* $\langle\!\langle v \rangle\!\rangle_G$ by

$$\langle\!\langle v \rangle\!\rangle_G = \bigvee_{f_\vee \in F_\vee(G)} \bigwedge \{val_G(Play_G(v)|_{f_\vee})\}$$

where, for $X \subseteq \mathcal{Z}$, $\bigvee X$ (resp. $\bigwedge X$) denotes the least upper bound (resp. greatest lower bound) of X. Thus, $\langle\!\langle v \rangle\!\rangle_G$ is the least upper bound to all play values the $\vee$-player can enforce. These definitions are analogous to the definitions in [18] for multi-valued model checking games. For infinite plays, we inherit the winning condition from parity games as considered, e.g., in [6, 19]. For the two-valued case (as well as for the finite-valued case in [18]), however, there exist optimal strategies for each player which are positional. As shown in the following example, this does not hold for crash games.

Example 1.



Consider the game $G = (V_\vee, V_\wedge, E, c, r)$ with $V_\vee = \{v\}$, $V_\wedge = \emptyset$, $E = \{(v, v), (v, \mathbf{0})\}$, $c((v, v)) = 1$, $c((v, \mathbf{0})) = 0$ and $r(v) = 2$ (cf. the fig.). The game value for v is ∞. This value, though, cannot be realized by any individual play. Instead there is, for every $z \in \mathbb{Z}$, a $\vee$-strategy f_z such that $val_G(\pi) = z$ for the single play $\pi \in Play_G(v)|_{f_z}$. For $z > 0$, this strategy, though, is not *positional*. □

Note that for the two-valued case, the algorithms and constructions heavily rely on the fact that there are optimal positional strategies for both players. For a crash game $G = (V_\vee, V_\wedge, E, c, r)$ we only have the remarkable property that the choice only depends on the current payoff and position. I.e., for a given $z \in \mathbb{Z}$ with $z \leq \langle\!\langle v \rangle\!\rangle_G$, there exists a $\vee$-strategy f_z with $\bigwedge \{val_G(Play_G(v)|_{f_z})\} \geq z$ and the property that $f_z(\pi v) = f_z(\pi' v)$ whenever $c(\pi v) = c(\pi' v)$.

3 Hierarchical Systems of Simple Integer Equations

In the next section, we will reduce the problem of computing game values of crash games to the problem of solving hierarchical systems of simple integer equations. An integer equation $\mathbf{x} = e$ is called *simple* iff the right-hand side e is of the form

$$e \ ::= \ c \mid \mathbf{x} \mid e + a \mid e_1 \wedge e_2 \mid e_1 \vee e_2$$

where $\mathbf{x}$ is a variable, e, e_1, e_2 are expressions, and $a, c \in \mathbb{Z}$. Note that we restrict addition such that the second argument is always a constant. These second arguments are called *addition constants* whereas other constants c are called *basic constants*. The operator $+a$ has highest precedence, followed by $\wedge$ and finally $\vee$ which has lowest precedence. A *system* $\mathcal{E}$ of simple integer equations is a finite sequence $\mathbf{x}_1 = e_1, \ldots, \mathbf{x}_n = e_n$ of simple integer equations where $\mathbf{x}_1, \ldots, \mathbf{x}_n$ are pairwise distinct variables. Let us denote the set of variables $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ occurring in $\mathcal{E}$ by $\mathbf{X}$. The system $\mathcal{E}$ is called *conjunctive* (*disjunctive*) iff no right-hand side contains the maximum-operator "$\vee$" (minimum-operator "$\wedge$"). For a *variable assignment* $\mu : \mathbf{X} \to \mathcal{Z}$ an expression e is mapped to a value $[\![e]\!]\mu \in \mathcal{Z}$ as follows:

$$[\![c]\!]\mu = c \qquad [\![\mathbf{x}]\!]\mu = \mu(\mathbf{x}) \qquad [\![a + e]\!]\mu = a + [\![e]\!]\mu$$
$$[\![e_1 \wedge e_2]\!]\mu = [\![e_1]\!]\mu \wedge [\![e_2]\!]\mu \qquad [\![e_1 \vee e_2]\!]\mu = [\![e_1]\!]\mu \vee [\![e_2]\!]\mu$$

where $\mathbf{x}$ is a variable, e, e_1, e_2 are expressions, and $a, c \in \mathbb{Z}$. Here, we extend the operation "$+$" to $\pm\infty$ by: $x + (-\infty) = (-\infty) + x = -\infty$ for all x and $x + \infty = \infty + x = \infty$ for all $x > -\infty$. Thus, "$+$" distributes over $\vee$ and $\wedge$. Assume that $\mathcal{E}$ denotes the system $\mathbf{x}_1 = e_1, \ldots, \mathbf{x}_n = e_n$. As usual, a *solution* of $\mathcal{E}$ is a variable assignment μ which satisfies all equations of $\mathcal{E}$, i.e. $\mu(\mathbf{x}_i) = [\![e_i]\!]\mu$ for $i = 1, \ldots, n$. We also use the term *fixpoint* instead of solution. We call a variable assignment μ a *pre-solution* of $\mathcal{E}$ iff $\mu(\mathbf{x}_i) \leq [\![e_i]\!]\mu$ for $i = 1, \ldots, n$ and a *post-solution* of $\mathcal{E}$ iff $\mu(\mathbf{x}_i) \geq [\![e_i]\!]\mu$ for $i = 1, \ldots, n$. Note that the function $[\![e]\!] : (\mathbf{X} \to \mathcal{Z}) \to \mathcal{Z}$ is monotonic for every expression e.

A *hierarchical* system $\mathcal{H} = (\mathcal{E}, r)$ of simple integer equations consists of a system $\mathcal{E}$ of simple integer equations and a rank function r mapping the variables $\mathbf{x}_i$ of $\mathcal{E}$ to natural numbers $r(\mathbf{x}_i) \in \{1, \ldots, d\}, d \in \mathbb{N}$. For variables with odd (resp. even) rank, we are interested in greatest (resp. least) solutions. Further, the variables of greater ranks are assumed to live within the scopes of the variables with smaller ranks. We call the resulting solution *canonical*. In order to define the canonical solution formally, let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $\mathbf{X}_{\bowtie j}$ denote the set of variables $\mathbf{x}_i$ with $r(\mathbf{x}_i) \bowtie j$ where $\bowtie \in \{=, <, >, \leq, \geq\}$. Then the equations $\mathbf{x}_i = e_i$ of $\mathcal{E}$ with $\mathbf{x}_i \in \mathbf{X}_{\geq j}$ define a monotonic mapping $[\![\mathcal{E}, r]\!]_j$ from the set $\mathbf{X}_{<j} \to \mathcal{Z}$ of variable assignments with domain $\mathbf{X}_{<j}$ into the set $\mathbf{X}_{\geq j} \to \mathcal{Z}$ of variable assignments with domain $\mathbf{X}_{\geq j}$. Assume that j is even, i.e., corresponds to a least solution. Given the mapping $[\![\mathcal{E}, r]\!]_{j+1}$, the mapping $[\![\mathcal{E}, r]\!]_j$ is given by:

$$[\![\mathcal{E}, r]\!]_j \ \rho = \mu + [\![\mathcal{E}, r]\!]_{j+1}(\rho + \mu)$$

where $\rho : \mathbf{X}_{<j} \to \mathcal{Z}$ is a variable assignment and $\mu : \mathbf{X}_{=j} \to \mathcal{Z}$ is the *least* variable assignment such that

$$\mu(\mathbf{x}_i) = [\![e_i]\!](\rho + \mu + [\![\mathcal{E}, r]\!]_{j+1}(\rho + \mu))$$

for all $\mathbf{x}_i \in \mathbf{X}_{=j}$. Here, the operator "$+$" denotes combination of two variable assignments with disjoint domains. The case where j is odd, i.e., corresponds to a greatest solution is analogous. Finally, the canonical solution μ^* is given by $[\![\mathcal{E}, r]\!]_1$ applied to the empty variable assignment $\{\}$. The next example illustrates how one can compute the canonical solution by a transfinite fixpoint iteration.

Example 2. Consider the system of equations

$$\mathbf{x}_1 = 5 + \mathbf{x}_2 \wedge 7 \qquad \mathbf{x}_2 = \mathbf{x}_3 \qquad \mathbf{x}_3 = -5 + \mathbf{x}_1 \vee 1$$

where $r(\mathbf{x}_1) = r(\mathbf{x}_2) = 1$ and $r(\mathbf{x}_3) = 2$. Thus $\mathbf{x}_3$ lives within the scope of $\mathbf{x}_1, \mathbf{x}_2$.

	0		1		2		3	
$\mathbf{x}_1$	∞		7		7		7	
$\mathbf{x}_2$	∞		∞		∞		2	
$\mathbf{x}_3$	0	1	0	1	0	1	0	1
	$-\infty$	∞	$-\infty$	∞	$-\infty$	2	$-\infty$	2

The fixpoint iteration is illustrated in the table at the right-hand side. The column labeled with i corresponds to the i-th outer iteration step. The inner iterations are illustrated by the tables in the row for the variables $\mathbf{x}_3$. As for the outer iteration, the column labeled with i contains the value after the i-th inner iteration step. Since we are interested in greatest solutions for the variables $\mathbf{x}_1$ and $\mathbf{x}_2$, the outer iteration starts with the value ∞ for these variables. Then, the inner iteration for $\mathbf{x}_3$ starts with $-\infty$ and reaches a fixpoint after one iteration step. Then, the outer iteration goes on with the new values for $\mathbf{x}_1$, $\mathbf{x}_2$ and $\mathbf{x}_3$. Finally, we get the canonical solution after three outer iteration steps. □

Note that in general transfinite fixpoint iterations are necessary for computing canonical solutions. Related systems over non-negative integers have been considered in [17]. Zero-one-valued systems using minimum and maximum only are also known as *Boolean* fixpoint equations and can be used for checking validity of propositional μ-calculus formulas interpreted over finite labeled transition systems or for computing the winning positions of parity games [1].

4 Computing Game Values

Instead of determining game values of crash games directly, we reduce this problem to solving hierarchical systems of simple integer equations. Although there is a one-to-one correspondence, we are here interested in the reduction from crash games to hierarchical systems only. Let $G = (V_\vee, V_\wedge, E, c, r)$ denote a crash game. We construct a corresponding system $\mathcal{E}_G$ of simple integer equations which uses variables from the set $\mathbf{X} = \{\mathbf{x}_v \mid v \in V_\vee \cup V_\wedge\}$ as follows. For each position $v \in V_\vee$, we add the equation

$$\mathbf{x}_v = \bigvee\nolimits_{v' \in \{v\}E} ([v'] + c(v, v'))$$

and, likewise, for each position $v \in V_\wedge$, we add the equation

$$\mathbf{x}_v = \bigwedge\nolimits_{v' \in \{v\}E} ([v'] + c(v, v'))$$

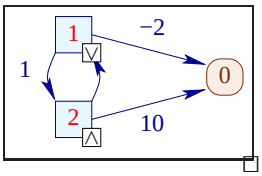
where $[v]$ denotes the variable $\mathbf{x}_v$ if $v \in V_\vee \cup V_\wedge$ and $[\mathbf{0}] = 0$. Then the *hierarchical* system $\mathcal{H}_G$ of simple integer equations which corresponds to the crash game G is the pair $(\mathcal{E}_G, r_G)$ where $r_G(\mathbf{x}_v) = r(v)$ for $v \in V_\vee \cup V_\wedge$.

Example 3.

The Fig. on the right shows a crash game with two positions, say, 1 and 2 of the respective ranks. Then the corresponding system of integer equations is given by

$$\mathbf{x}_1 = \mathbf{x}_2 + 1 \vee -2 \qquad \mathbf{x}_2 = \mathbf{x}_1 \wedge 10$$

where the rank of $\mathbf{x}_i$ equals i.



□

Theorem 1. *For a crash game $G = (V_\vee, V_\wedge, E, c, r)$, let μ^* denote the canonical solution of $\mathcal{H}_G$. Then $\langle\!\langle v \rangle\!\rangle_G = \mu^*(\mathbf{x}_v)$ for all $v \in V_\vee \cup V_\wedge$. Furthermore $\mathcal{H}_G$ can be constructed in time $\mathcal{O}(|V_\vee \cup V_\wedge| + |E|)$. Vice-versa, given a hierarchical system of equations $\mathcal{H}$ we can compute a crash game $G = (V_\vee, V_\wedge, E, c, r)$ in time $\mathcal{O}(|\mathcal{E}|)$ whose game values corresponds to the values of the canonical solution of $\mathcal{H}$.* □

Note that theorem 1 also holds if we define $\langle\!\langle v \rangle\!\rangle_G$ as $\bigwedge_{f_\wedge \in F_\wedge(G)} \bigvee \{ val_G(\pi) \mid \pi \in Play_{G,v}|_{f_\wedge} \}$. Thus, we get the following duality theorem as a corollary:

Theorem 2. *Let $G = (V_\vee, V_\wedge, E, c, r)$ be a crash game and $v \in V_\vee \cup V_\wedge \cup \{\mathbf{0}\}$. Then $\langle\!\langle v \rangle\!\rangle_G = \bigwedge_{f_\wedge \in F_\wedge(G)} \bigvee \{ val_G(Play_G(v)|_{f_\wedge}) \}$.* □

5 Solving Hierarchical Systems

In this section, we present our strategy improvement algorithm for computing canonical solutions. Assume that $\mathcal{H} = (\mathcal{E}, r)$ is a hierarchical system of simple equations where the range of r is contained in the set $\{1, \ldots, d\}$. Instead of solving the original system over $\mathcal{Z}$, we consider a corresponding system over an instrumented lattice. In case that all solutions of this system are finite, the instrumentation will assure that the canonical solution is the only solution. The instrumentation technique here is a generalization of the instrumentation used in [8] to determine *least solutions* of systems of integer equations. We instrument $\mathcal{Z}$ by introducing one extra component from $\mathbb{N}$ for every $j = 1, \ldots, d$. Thus, we consider the instrumented lattice $\mathcal{D}_d = \mathbb{D}_d \cup \{-\infty, \infty\}$ with $\mathbb{D}_d = \mathbb{Z} \times \mathbb{N}^d$ where $-\infty$ is the least and ∞ is the greatest element and the ordering on $\mathbb{D}_d$ (the *finite* elements of $\mathcal{D}_d$) is given by:

$$(a, j_1, \ldots, j_d) < (a', j'_1, \ldots, j'_d)$$

iff $a < a'$ or $a = a'$ and there exists some $1 \leq k \leq d$ with the following properties:

1. $j_i = j'_i$ for all $i < k$;
2. $j_k > j'_k$ whenever k is even;
3. $j_k < j'_k$ whenever k is odd.

Note that values get larger w.r.t. this ordering when components corresponding to greatest fixpoints are increased or components corresponding to least fixpoints are decreased. Addition on the finite elements of $\mathcal{D}_d$ operates on all components simultaneously, i.e.:

$$(a, j_1, \ldots, j_d) + (a', j'_1, \ldots, j'_d) = (a + a', j_1 + j'_1, \ldots, j_d + j'_d)$$

Note that $+$ distributes over $\vee$ and $\wedge$. The evaluation of an expression e over $\mathcal{D}_d$ will be denoted by $[\![e]\!]^\sharp$. As for expressions over $\mathcal{Z}$, $[\![e]\!]^\sharp$ is monotonic.

A slightly different choice is made in [8] where an extra operator inc is introduced for incrementing the extra component. Accordingly, our *lifting* transformation differs from the one chosen in [8]. In order to *lift* an equation $\mathbf{x}_i = e_i$ to $\mathcal{D}_d$, we replace every finite constant $c \in \mathbb{Z}$ occurring in e_i with $(c, 0, \ldots, 0)$. Moreover, we replace every equation $\mathbf{x}_i = e_i$ with $\mathbf{x}_i = e_i + \mathbf{1}_{r(\mathbf{x}_i)}$ where $\mathbf{1}_k$ is the $(d+1)$-tuple consisting of 0 everywhere besides the $(k+1)$-th component where it equals 1. We denote the lifted system of simple integer equations by $\mathcal{E}^\sharp$. To simplify notations we define $\beta : \mathcal{D}_d \to \mathcal{Z}$ by

$$\beta(-\infty) = -\infty \qquad \beta(\infty) = \infty \qquad \beta(z, j_1, \ldots, j_d) = z.$$

The following theorem states that we can recover the canonical solution of the original system from the canonical solution of the corresponding lifted system.

Theorem 3. *Assume that $(\mathcal{E}, r)$ is a hierarchical system. Let $\mathcal{E}^\sharp$ be the lifted system corresponding to $\mathcal{E}$ and let $\mu^\sharp$ be the canonical solution of the hierarchical system $(\mathcal{E}^\sharp, r)$. Then $\beta \circ \mu^\sharp$ is the canonical solution of $(\mathcal{E}, r)$.* □

Our key observation is that *finite* solutions of lifted systems are unique. Here, a variable assignment μ is called *finite* iff $-\infty < \mu(\mathbf{x}_i) < \infty$ for all variables $\mathbf{x}_i$. For an equation system $\mathcal{E}$, let $a_\mathcal{E}$ denote the sum of the smallest basic constant together with all negative addition constants. Moreover, let $b_\mathcal{E}$ denote the sum of the largest basic constant together with all positive addition constants. We have:

Theorem 4. *Assume that $\mathcal{E}^\sharp$ is the system of lifted equations corresponding to the hierarchical system $(\mathcal{E}, r)$ with n variables where n_k variables are of rank k. Then:*

1. *$\mathcal{E}^\sharp$ has at most one finite solution.*
2. *If $-\infty < \mu(\mathbf{x}_i) < \infty$ for a solution μ of $\mathcal{E}^\sharp$ and a variable $\mathbf{x}_i$, then $\mu(\mathbf{x}_i) = (a, j_1, \ldots, j_d)$ for some $a \in [a_\mathcal{E}, b_\mathcal{E}]$ and $j_1, \ldots, j_d \in \mathbb{N}$ with $0 \le j_k \le n_k$.*
3. *If $\mathcal{E}^\sharp$ is conjunctive, the greatest solution of $\mathcal{E}^\sharp$ can be computed in time $\mathcal{O}(d \cdot n \cdot |\mathcal{E}|)$.*

Proof. To simplify the proof, here we additionally allow the constants $-\infty$ and ∞ to occur as basic constants. In order to prove assertion 1, we first consider the case of a lifted system which consists in exactly one equation. W.l.o.g. consider the equation

$$\mathbf{x} = \mathbf{x} + a \wedge b \vee c$$

where $(0, \ldots, 0) \neq a \in \mathbb{D}_d$ and $b, c \in \mathcal{D}_d$. Assume that μ is a finite solution of the above system. We show that μ is given by

$$\mu(\mathbf{x}) \;=\; \begin{cases} c & \text{if } a < (0, \ldots, 0) \\ b \vee c & \text{if } a > (0, \ldots, 0). \end{cases}$$

Note that necessarily $c \le \mu(\mathbf{x}) \le b \vee c$. If $b \le c$, the statement follows immediately. Assume therefore that $b > c$. First assume that $a > (0, \ldots, 0)$ and $c \le \mu(\mathbf{x}) < b \vee c = b$. Then $\mu(\mathbf{x}) = [\![\mathbf{x} + a]\!]^\sharp \mu$ — which is impossible for finite values. Now assume that $a < (0, \ldots, 0)$ and $c < \mu(\mathbf{x}) \le b \vee c = b$. Then $\mu(\mathbf{x}) = [\![\mathbf{x} + a]\!]^\sharp \mu$ — which is again impossible for finite values.

Now we consider the general case. We show assertion 1 and 2 simultaneously. Therefore, we first introduce the following notations. Let $\mathcal{E}^\sharp$ denote a system of

equations over $\mathcal{D}_d$. We call a sequence π of expressions $e_1, \ldots, e_k$ a *path* iff for $i = 1, \ldots, k-1$ either e_i is a variable $\mathbf{x}_j$ and $\mathbf{x}_j = e_{i+1}$ is an equation of $\mathcal{E}^\sharp$ or e_{i+1} is an immediate subexpression of e_i. The path π is *short* iff all expressions e_i that are variables are distinct. In order to define the weight of a system of simple integer equations, we set $w(e) = a$, if e denotes an expression $e' + a$, $w(e) = c$, if e denotes an expression c, and, $w(e) = 0$ otherwise. Thereby e, e' denote expressions, a denotes an addition constant and c a basic constant. Then, the sum $\sum_{i=1,\ldots,k} w(e_i)$ is called the *weight* of the path. Let P denote the set of all short paths ending with a *finite* basic constant. We define $w_{max}(\mathcal{E}^\sharp)$ (resp. $w_{min}(\mathcal{E}^\sharp)$) as the maximal (resp. minimal) weight of paths in P. Furthermore, for $j = 1, \ldots, d$, we define $w_j(\mathcal{E}^\sharp)$ to be the maximum of the $j+1$-th component of the weights of paths in P. We call $[w_{min}, w_{max}]$ and w_j for $j = 1, \ldots, d$ the weights of $\mathcal{E}^\sharp$.

Let $\mathcal{E}^\sharp$ be the lifted system $\mathbf{x}_1 = e_1, \ldots, \mathbf{x}_n = e_n$. Assume that μ is a finite solution of $\mathcal{E}^\sharp$. We show by induction on the number of variables occurring in right-hand sides that the following holds:

1. μ is the only finite solution of $\mathcal{E}^\sharp$;
2. $\mu(\mathbf{x}) \in [w_{min}(\mathcal{E}^\sharp), w_{max}(\mathcal{E}^\sharp)]$ for every variable $\mathbf{x}$ and
3. the $(j+1)$-th component of $\mu(\mathbf{x})$ is less than or equal to $w_j(\mathcal{E}^\sharp)$.

The statement is obviously fulfilled if there are no variables in right-hand sides. For the induction step let $\mathbf{x}_i$ be a variable that occurs in a right-hand side of $\mathcal{E}^\sharp$, and consider the equation $\mathbf{x}_i = e_i$ of $\mathcal{E}^\sharp$.

If e_i does not contain $\mathbf{x}_i$, we can substitute e_i everywhere in the remaining equations for $\mathbf{x}_i$ to obtain a system $\mathcal{E}^{\sharp\prime}$ with the same set of solutions and the same weights. Since $\mathbf{x}_i$ does not occur in right-hand sides any more, the result follows by the induction hypothesis.

Otherwise, we first have to replace the equation $\mathbf{x}_i = e_i$ by an equation $\mathbf{x}_i = e$ s.t. (1) e does not contain the variable $\mathbf{x}_i$ and (2) the systems $\mathbf{x}_i = e_i\sigma$ and $\mathbf{x}_i = e\sigma$ have the same set of finite solutions for every substitution σ mapping variables other than $\mathbf{x}_i$ to finite values. Then replacing $\mathbf{x}_i = e_i$ by $\mathbf{x}_i = e$ will preserve the set of finite solutions. A suitable expression e is constructed as follows. By using distributivity, we rewrite the equation $\mathbf{x}_i = e_i$ into

$$\mathbf{x}_i = \mathbf{x}_i + a \wedge e_1' \vee e_2'$$

where $\mathbf{x}_i + a \wedge e_1' \vee e_2'$ is in disjunctive normal form. Given e' as $e_1' \vee e_2'$ if $a > (0, \ldots, 0)$ and as e_2' if $a < (0, \ldots, 0)$, we get, from our considerations for a single equation, that the systems $\mathbf{x}_i = e_i\sigma$ and $\mathbf{x}_i = e'\sigma$ (which consist of a single equation) have the same set of finite solutions for every substitution σ mapping variables other than $\mathbf{x}_i$ to finite values. Since $e_1' \vee e_2'$ and e_2' are in disjunctive normal form and have one occurrence of the variable $\mathbf{x}_i$ less than e, this process can be repeated to eliminate all occurrences of the variable $\mathbf{x}_i$. Doing so, an expression e with the desired properties can be obtained. Thus, we can replace the equation $\mathbf{x}_i = e_i$ with $\mathbf{x}_i = e$ and substitute every occurrence of $\mathbf{x}_i$ in right-hand sides with e to obtain a system $\mathcal{E}^{\sharp\prime}$ of equations which has the same set of finite solutions as $\mathcal{E}^\sharp$. Furthermore the weights of $\mathcal{E}^{\sharp\prime}$ are less than or equal to the

weights of $\mathcal{E}^\sharp$. Since $\mathbf{x}_i$ does not occur in a right-hand side of $\mathcal{E}^{\sharp\prime}$, we can apply the induction hypothesis to finish the proof.

The above implies assertion 1. Since

$$[w_{min}(\mathcal{E}^\sharp), w_{max}(\mathcal{E}^\sharp)] \subseteq [(a_\mathcal{E}, 0, n_2, \ldots), (b_\mathcal{E}, n_1, 0, \ldots)]$$

and $w_j(\mathcal{E}^\sharp) \leq n_j$ for $j = 1, \ldots, d$ assertion 2 follows for finite solutions. Non-finite solutions can be reduced to the finite case by removing all infinite variables. The third assertion holds, since, by similar arguments as in [7], n rounds of Round-Robin iteration suffice to compute μ. Since elements in $\mathbb{D}_d$ are $(d+1)$-tuples, addition and comparison has uniform complexity $\mathcal{O}(d)$. □

In particular, theorem 4 implies that finite values in solutions are *bounded*:

Corollary 1. *Assume that μ^* denotes the canonical solution of a hierarchical system $(\mathcal{E}, r)$ over $\mathcal{Z}$. Then (1) $\mu^*(\mathbf{x}_i) = -\infty$ whenever $\mu^*(\mathbf{x}_i) < a_\mathcal{E}$ and (2) $\mu^*(\mathbf{x}) = \infty$ whenever $\mu^*(\mathbf{x}_i) > b_\mathcal{E}$.* □

Note that this corollary has important consequences for crash games. It implies that every finite game value lies in the interval $[a_\mathcal{E}, b_\mathcal{E}]$.

Now assume that the canonical solution μ^* of the hierarchical system $(\mathcal{E}, r)$ over $\mathcal{Z}$ and hence also the canonical solution $\mu^\sharp$ of the corresponding lifted hierarchical system $(\mathcal{E}^\sharp, r)$ is finite and thus by theorem 4 the only finite solution. By theorem 3 our problem of computing μ^* reduces to the computation of $\mu^\sharp$. Assume that $\mathcal{E}^\sharp$ consists of the equations $\mathbf{x}_i = e_i, i = 1, \ldots, n$, and let $a^\sharp_\mathcal{E}$ and $b^\sharp_\mathcal{E}$ denote the corresponding lifted constants:

$$a^\sharp_\mathcal{E} = (a_\mathcal{E}, 0, n_2, 0, n_4 \ldots) \qquad b^\sharp_\mathcal{E} = (b_\mathcal{E}, n_1, 0, n_3, 0, \ldots)$$

where n_k is the number of variables of rank k. In order to compute $\mu^\sharp$, we replace each equation $\mathbf{x}_i = e_i$ of $\mathcal{E}^\sharp$ with $\mathbf{x}_i = e_i \wedge b^\sharp_\mathcal{E} \vee a^\sharp_\mathcal{E}$. For simplicity, we denote the resulting system again by $\mathcal{E}^\sharp$. Since $\mathcal{E}^\sharp$ does not have non-finite solutions any more, by theorem 4, $\mu^\sharp$ is now the *only* solution of $\mathcal{E}^\sharp$. In order to compute $\mu^\sharp$ we propose *strategy iteration*.

Let $M_\vee(\mathcal{E}^\sharp)$ denote the set of all $\vee$-expressions in $\mathcal{E}^\sharp$. A *strategy* π is a function mapping every expression $e_1 \vee e_2$ in $M_\vee(\mathcal{E}^\sharp)$ to one of the subexpressions e_1, e_2. Given a strategy π together with an expression e, we write $e\pi$ for the expression obtained by recursively replacing every $\vee$-expression in $\mathcal{E}^\sharp$ by the respective subexpression selected by π. Formally, $e\pi$ is given as:

$$\begin{array}{llll} c\pi & = c & (e_1 \wedge e_2)\pi = e_1\pi \wedge e_2\pi & \mathbf{x}_i\pi = \mathbf{x}_i \\ (e+a)\pi & = e\pi + a\pi & (e_1 \vee e_2)\pi = (\pi(e_1 \vee e_2))\pi & \end{array}$$

Accordingly, the system $\mathcal{E}^\sharp(\pi)$ of equations extracted from $\mathcal{E}^\sharp$ via the strategy π is the system $\mathbf{x}_i = e_i\pi$, $i = 1, \ldots, n$, assuming that $\mathcal{E}^\sharp$ is the system $\mathbf{x}_i = e_i$, $i = 1, \ldots, n$. $\mathcal{E}^\sharp(\pi)$ is a conjunctive system.

Assume that $\mu^\sharp_\pi$ denotes the greatest solution of $\mathcal{E}^\sharp(\pi)$ for a strategy π. By monotonicity, $\mu^\sharp_\pi \leq \mu^\sharp$ for all strategies π. Given a strategy π and the greatest solution $\mu^\sharp_\pi$ of $\mathcal{E}^\sharp(\pi)$ our goal is to determine an improved strategy π' such that the greatest solution

Algorithm 1 Strategy Improvement Algorithm

```
/* The system E^♯ has only finite solutions. */
μ ← variable assignment which maps every variable to −∞;
while (μ is not a solution of E^♯) {
    π ← P(μ);
    μ ← greatest solution of E^♯(π);
}
return μ;
```

$\mu^\sharp_{\pi'}$ of $\mathcal{E}^\sharp(\pi')$ is nearer to $\mu^\sharp$ than $\mu^\sharp_\pi$, i.e. $\mu^\sharp_\pi < \mu^\sharp_{\pi'} \leq \mu^\sharp$. Thereby we pursue the policy to modify π at all expressions $e = e_1 \vee e_2$ where $[\![\pi(e)]\!]^\sharp \mu^\sharp_\pi \neq [\![e]\!]^\sharp \mu^\sharp_\pi$ simultaneously. Therefore, we define the strategy $P(\mu)$ induced by a variable assignment μ by:

$$P(\mu)(e_1 \vee e_2) = \begin{cases} e_1 & \text{if } [\![e_1]\!]^\sharp \mu \geq [\![e_2]\!]^\sharp \mu \\ e_2 & \text{if } [\![e_1]\!]^\sharp \mu < [\![e_2]\!]^\sharp \mu \end{cases}$$

The following lemma implies that we can consider $P(\mu^\sharp_\pi)$ as an improvement of π.

Lemma 1. *Let $\mu^\sharp$ denote the only solution of the system $\mathcal{E}^\sharp$. Let $\mu < \mu^\sharp$ be a pre-solution of $\mathcal{E}^\sharp$. Let μ' denote the greatest solution of $\mathcal{E}^\sharp(P(\mu))$. Then $\mu < \mu'$.*

Proof. Since $\mu^\sharp$ is the only solution of $\mathcal{E}^\sharp$, μ is no solution of $\mathcal{E}^\sharp$. By the definition of P, μ is also a pre-solution of $\mathcal{E}^\sharp(P(\mu))$ and no solution of $\mathcal{E}^\sharp(P(\mu))$. Since μ is a pre-solution, Knaster-Tarski's fixpoint theorem implies that $\mu \leq \mu'$. Moreover, $\mu \neq \mu'$, since μ is no solution. □

According to lemma 1, we can compute $\mu^\sharp$ using alg. 1. For the correctness of alg. 1 consider the following argumentation. Obviously, alg. 1 returns the unique solution $\mu^\sharp$ of $\mathcal{E}^\sharp$ whenever it terminates. Let $\pi_1, \pi_2, \ldots$ denote the sequence of occurring strategies. Since the program variable μ is always the greatest solution of $\mathcal{E}^\sharp(\pi)$ for some strategy π, μ is always a pre-solution of $\mathcal{E}^\sharp$ and $\mu \leq \mu^\sharp$. Therefore, by lemma 1, the greatest solutions $\mu^\sharp_{\pi_i}$ of $\mathcal{E}^\sharp(\pi_i)$ form a strictly increasing sequence. In particular no strategy occurs twice in the sequence $\pi_1, \pi_2, \ldots$. Since the total number of strategies is bounded, the algorithm eventually terminates. For a precise characterization of the run-time, let $\Pi(m)$ denote the maximal number of updates of strategies necessary for systems with m $\vee$-expressions. We have:

Theorem 5. *Assume that $(\mathcal{E}, r)$ is hierarchical system with n variables and m $\vee$-expressions where the canonical solution μ^* of $(\mathcal{E}, r)$ is finite. Then μ^* can be computed by strategy iteration in time $\mathcal{O}(d \cdot n \cdot |\mathcal{E}| \cdot \Pi(m+n))$.* □

The following example illustrates a run of alg. 1.

Example 4. Consider the system $(\mathcal{E}, r)$ given by:

$$\mathbf{x}_1 = \mathbf{x}_2 + -1 \wedge 10 \qquad \mathbf{x}_2 = \mathbf{x}_1 \wedge \mathbf{x}_2 + 1 \vee 0$$

where $r(\mathbf{x}_1) = 1$ and $r(\mathbf{x}_2) = 2$. The canonical solution maps $\mathbf{x}_1$ to -1 and $\mathbf{x}_2$ to 0. The corresponding lifted system $\mathcal{E}^\sharp$ is given by

$$\begin{aligned} \mathbf{x}_1 &= ((\mathbf{x}_2 + (-1,0,0)) \wedge (10,0,0)) + (0,1,0)) \wedge (11,1,0) \vee (-1,0,1) \\ \mathbf{x}_2 &= ((\mathbf{x}_1 \wedge \mathbf{x}_2 + (1,0,0) \vee (0,0,0)) + (0,0,1)) \wedge (11,1,0) \vee (-1,0,1) \end{aligned}$$

where we already have added the safe lower and upper bounds. After the first iteration, the value of the program variable μ is the variable assignment μ_0 mapping every variable to the lower bound $(-1, 0, 1)$. The resulting strategy $P(\mu_0)$ gives as the system:

$$\mathbf{x}_1 = (-1, 0, 1) \qquad \mathbf{x}_2 = (0, 0, 0) + (0, 0, 1) \wedge (11, 1, 0)$$

with the obvious greatest solution. Accordingly, the next improvement results in:

$$\begin{aligned} \mathbf{x}_1 &= ((\mathbf{x}_2 + (-1, 0, 0)) \wedge (10, 0, 0)) + (0, 1, 0) \wedge (11, 0, 0) \\ \mathbf{x}_2 &= (0, 0, 0) + (0, 0, 1) \wedge (11, 1, 0) \end{aligned}$$

giving us the unique finite solution of $\mathcal{E}^\sharp$ that corresponds to the canonical solution. □

The efficiency of strategy iteration crucially depends on the size of the factor $\Pi(m)$. In practical implementations this factor seems to be surprisingly small. Interestingly, though, it is still open whether (or: under which circumstances) the trivial upper bound of 2^m for $\Pi(m)$ can be significantly improved [19, 2].

6 General Canonical Solutions

In this section we show how the restriction to finite canonical solutions can be lifted. The idea is to successively identify variables which are $-\infty$ or ∞ in the canonical solution and to remove these from the system until the remaining system has a finite canonical solution. Let $\mathcal{B} = \{0 < 1\}$ denote the Boolean lattice, and consider the mappings $\alpha_{-\infty} : \mathcal{Z} \to \mathcal{B}$ and $\alpha_\infty : \mathcal{Z} \to \mathcal{B}$ defined by:

$$\begin{array}{llll} \alpha_{-\infty}(-\infty) = 0 & \alpha_{-\infty}(z) = 1 & \text{if } z > -\infty \\ \alpha_\infty(\infty) \quad = 1 & \alpha_\infty(z) \;\; = 0 & \text{if } z < \infty \end{array}$$

The mapping $\alpha_{-\infty}$ commutes with arbitrary least upper bounds whereas the mapping α_∞ commutes with arbitrary greatest lower bounds. Additionally, we have:

$$\begin{array}{ll} \alpha_{-\infty}(x + a) = \alpha_{-\infty}(x) & \alpha_\infty(x + a) = \alpha_\infty(x) \\ \alpha_{-\infty}(x \vee y) = \alpha_{-\infty}(x) \vee \alpha_{-\infty}(y) & \alpha_\infty(x \vee y) = \alpha_\infty(x) \vee \alpha_\infty(y) \\ \alpha_{-\infty}(x \wedge y) = \alpha_{-\infty}(x) \wedge \alpha_{-\infty}(y) & \alpha_\infty(x \wedge y) = \alpha_\infty(x) \wedge \alpha_\infty(y) \end{array}$$

where $x, y \in \mathcal{Z}$ and $a \in \mathbb{Z}$. Thus, the mappings $\alpha_{-\infty}$ and α_∞ are homomorphisms mapping the operations "$+\,a$", "$\wedge$", and "$\vee$" on $\mathcal{Z}$ to logical connectivities. Using these abstractions we define, for a system $\mathcal{E}$ of equations over $\mathcal{Z}$, the system $\mathcal{E}_{-\infty}$ of equations over $\mathcal{B}$ as the system obtained from $\mathcal{E}$ by applying $\alpha_{-\infty}$ to all constants and substituting the operators accordingly. Analogously, we define the system $\mathcal{E}_\infty$ of equations over $\mathcal{B}$ using the abstraction α_∞. The following lemma enables us to identify some variables that are $-\infty$ or ∞ in the canonical solution.

Lemma 2. *Assume $(\mathcal{E}, r)$ is a hierarchical equation system over $\mathcal{Z}$ with canonical solution μ^*. Let $\mu^*_{-\infty}$ and μ^*_∞ denote the canonical solutions of $(\mathcal{E}_{-\infty}, r)$ and $(\mathcal{E}_\infty, r)$, respectively. Then (1) $\mu^*(\mathbf{x}_i) = -\infty$ whenever $\mu^*_{-\infty}(\mathbf{x}_i) = 0$ and (2) $\mu^*(\mathbf{x}_i) = \infty$ whenever $\mu^*_\infty(\mathbf{x}_i) = 1$.* □

In order to deal with the second source of infinities we introduce the following notions. For $z \in \mathcal{Z}$, we write $(\mathcal{E}^{\vee z}, r)$ for the system obtained from $(\mathcal{E}, r)$ by replacing $\mathbf{x}_i = e_i$ with $\mathbf{x}_i = e_i \vee z$ for every variable $\mathbf{x}_i$ with odd rank. Accordingly, we write $(\mathcal{E}^{\wedge z}, r)$ for the system obtained from $(\mathcal{E}, r)$ by replacing $\mathbf{x}_i = e_i$ with $\mathbf{x}_i = e_i \wedge z$ for every variable $\mathbf{x}_i$ with even rank. We have:

Lemma 3. *Consider a hierarchical system $(\mathcal{E}, r)$ of integer equations. Assume that μ^* denotes the canonical solution of $(\mathcal{E}, r)$, and that $\mu_{a_\mathcal{E}}$ and $\mu_{b_\mathcal{E}}$ denote the canonical solutions of $(\mathcal{E}^{\vee a_\mathcal{E}-1}, r)$ and $(\mathcal{E}^{\wedge b_\mathcal{E}+1}, r)$, respectively. Then:*

1. *$\mu^* \leq \mu_{a_\mathcal{E}}$ and $\mu_{b_\mathcal{E}} \leq \mu^*$;*
2. *$\mu_{a_\mathcal{E}} = \mu^*$ whenever $\mu_{a_\mathcal{E}}(\mathbf{x}_i) \geq a$ for all variables $\mathbf{x}_i$ of odd rank;*
3. *$\mu_{b_\mathcal{E}} = \mu^*$ whenever $\mu_{b_\mathcal{E}}(\mathbf{x}_i) \leq b$ for all variables $\mathbf{x}_i$ $\mathbf{x}_i$ of even rank.* □

In order to compute the solution of a hierarchical system of simple integer equations, we proceed in two steps. First, we only consider systems with canonical solutions μ^* which never return $-\infty$, i.e. $\mu^*(\mathbf{x}_i) > -\infty$ for every variable $\mathbf{x}_i$.

Theorem 6. *Assume that $(\mathcal{E}, r)$ is a hierarchical system of simple integer equations with n variables and m occurrences of "$\vee$" where the canonical solution μ^* never returns $-\infty$. Then μ^* can be computed in time $\mathcal{O}(d \cdot n^2 \cdot |\mathcal{E}| \cdot \Pi(m+n))$.*

Proof. Assume that $\mathcal{E}$ consists of the equations $\mathbf{x}_i = e_i$ for $i = 1, \ldots, n$. Let μ_∞ denote the canonical solution of $(\mathcal{E}_\infty, r)$. In the first stage, we remove all variables $\mathbf{x}_i$ with $\mu_\infty(\mathbf{x}_i) = 1$. That means that we obtain a system $\mathcal{E}'$ from $\mathcal{E}$ by removing all equations $\mathbf{x}_i = e_i$ s.t. $\mu_\infty(\mathbf{x}_i) = 1$ and replacing all occurrences of these variables in right-hand sides by the constant ∞. The hierarchical system $(\mathcal{E}', r)$ is equivalent to $(\mathcal{E}, r)$ and moreover ${(\mathcal{E}')}_\infty, r)$ is equivalent to $(\mathcal{E}_\infty, r)$ for the remaining variables.

For the second stage, assume w.l.o.g. that for all variables $\mathbf{x}_i$ of $(\mathcal{E}, r)$, $\mu_\infty(\mathbf{x}_i) = 0$. Now let $b := b_\mathcal{E}$ denote the upper bound of $\mathcal{E}$ for finite values. Then by corollary 1 $\mu^*(\mathbf{x}_i) \leq b$, if $\mu^*(\mathbf{x}_i) \in \mathbb{Z}$. Since the canonical solution μ_b of $(\mathcal{E}^{\wedge b+1}, r)$ is finite, it can be computed by strategy iteration according to theorem 5.

If $\mu_b(\mathbf{x}_i) \leq b$ for all variables $\mathbf{x}_i$ we are done, since lemma 3 implies that $\mu^* = \mu_b$. Otherwise, $\mu_b(\mathbf{x}_i) > b$ for some variable $\mathbf{x}_i$ which (using corollary 1) implies that $\mu^*(\mathbf{x}_i) = \infty$. Then we can again remove the equation for $\mathbf{x}_i$ and replace all occurrences of the variable $\mathbf{x}_i$ in right-hand sides of the remaining equations by ∞ to obtain an equivalent system.

Repeating this two-stage procedure for the resulting system with fewer variables may identify more variables of value ∞ until either all variables are found to obtain values ∞ or have values at most b. Then we have found the canonical solution μ^*. Summarizing, we apply the sub-routine for finite canonical systems at most n times for computing the abstractions and another n times for the bounded versions of the integer system — giving us the complexity statement. □

In the second step, we also lift the restriction that canonical solutions should never return $-\infty$. The key idea is now to repeatedly use the abstraction $\alpha_{-\infty}$ together with the algorithm from theorem 6.

Theorem 7. *Assume that $(\mathcal{E}, r)$ is a hierarchical system of simple integer equations with n variables and m occurrences of the maximum operator. Then the canonical solution can be computed in time $\mathcal{O}(d \cdot n^3 \cdot |\mathcal{E}| \cdot \Pi(m+n))$.* □

Proof. Assume that $\mathcal{E}$ consists of the equations $\mathbf{x}_i = e_i$ for $i = 1, \ldots, n$, and that μ^* is the canonical solution of $(\mathcal{E}, r)$. Let $\mu_{-\infty}$ denote the canonical solution of $(\mathcal{E}_{-\infty}, r)$. In the first stage, we remove all variables $\mathbf{x}_i$ with $\mu_{-\infty}(\mathbf{x}_i) = 0$. In the second stage therefore, $\mu_{-\infty}(\mathbf{x}_i) = 1$ for all variables $\mathbf{x}_i$. Let further $a = a_{\mathcal{E}}$ denote the lower bound of finite values in the canonical solution of $(\mathcal{E}, r)$. Let $\mu_{a_{\mathcal{E}}}$ denote the canonical solution of $(\mathcal{E}^{\vee a_{\mathcal{E}}-1}, r)$. By construction, $\mu_{a_{\mathcal{E}}}$ never returns $-\infty$. Thus we can apply theorem 6 to compute $\mu_{a_{\mathcal{E}}}$. Analogous as in the proof for theorem 6, we compute $\mu_{a_{\mathcal{E}}}$. This will provide us either with some new variables $\mathbf{x}_i$ with $\mu^*(\mathbf{x}_i) = -\infty$ or verify that $\mu_{a_{\mathcal{E}}}$ already equals μ^*. This two-stage procedure will be repeated until we have found the canonical solution.

Thus, similar to the proof of theorem 6 we repeatedly remove variables for which $\mu_{-\infty}$ returns 0 followed by the computation of the canonical solution of a hierarchical system whose canonical solution never returns $-\infty$. Since by theorem 6, the complexity of the corresponding sub-routine is $\mathcal{O}(d \cdot n^2 \cdot |\mathcal{E}| \cdot \Pi(m'))$ for $m' = m+n$, the assertion follows. □

Thus, using theorem 1 we have deduced our main theorem for crash games:

Theorem 8. *Assume that $G = (V_\vee, V_\wedge, E, c, r)$ is a crash game. Let $V = V_\vee \cup V_\wedge$. The values $\langle\!\langle v \rangle\!\rangle_G$, $v \in V$ can be computed by a strategy improvement algorithm in time $\mathcal{O}(d \cdot |V|^3 \cdot |G| \cdot \Pi(|G|))$, where $|G| = |V| + |E|$ and d denotes the maximal rank of a position occurring in G.* □

7 Conclusion

In this paper, we have introduced the concept of crash games where each player aims at optimizing the total payoff of a play. Although crash games do not admit optimal positional strategies, we succeeded in characterizing their game values by canonical solutions of hierarchical systems of simple integer equations.

We then have shown how the canonical solution of a hierarchical system of simple integer equations can be computed. For that, we used an *instrumentation* of the underlying lattice to obtain a lifted system of equations where finite solutions are *unique*. We exploited this insight to design a strategy iteration algorithm for hierarchical systems with *finite* canonical solutions. This algorithm is quite natural and comparable in its efficiency with the discrete strategy iteration algorithm for the Boolean case [19]. We then showed how general hierarchical systems of simple integer equations can be solved by iteratedly solving systems with finite canonical solutions only.

Using our algorithm for hierarchical systems, we are thus able to determine the game values of crash games. Further investigations are needed for automatically designing strategies with guaranteed payoffs.

References

1. André Arnold and Damina Niwinski. *Rudiments of μ-Calculus*, volume 146 of *Studies in Logic and The Foundations of Computer Science*. North-Holland, 2001.
2. Henrik Bjorklund, Sven Sandberg, and Sergei Vorobyov. Complexity of Model Checking by Iterative Improvement: the Pseudo-Boolean Framework . In *Proc. 5th Int. Andrei Ershov Memorial Conf. Perspectives of System Informatics*, pages 381–394. LNCS 2890, Springer, 2003.
3. Krishnendu Chatterjee, Thomas A. Henzinger, and Marcin Jurdzinski. Mean-payoff parity games. In *LICS*, pages 178–187. IEEE Computer Society, 2005.
4. Marsha Chechik, Benet Devereux, Steve M. Easterbrook, and Arie Gurfinkel. Multi-valued symbolic model-checking. *ACM Trans. Softw. Eng. Methodol.*, 12(4):371–408, 2003.
5. Alexandru Costan, Stephane Gaubert, Eric Goubault, Matthieu Martel, and Sylvie Putot. A Policy Iteration Algorithm for Computing Fixed Points in Static Analysis of Programs. In *Computer Aided Verification, 17th Int. Conf. (CAV)*, pages 462–475. LNCS 3576, Springer Verlag, 2005.
6. E. Allen Emerson and Charanjit S. Jutla. Tree automata, mu-calculus and determinacy (extended abstract). In *FOCS*, pages 368–377. IEEE, 1991.
7. Thomas Gawlitza, Jan Reineke, Helmut Seidl, and Reinhard Wilhelm. Polynomial Exact Interval Analysis Revisited. Technical report, TU München, 2006.
8. Thomas Gawlitza and Helmut Seidl. Precise fixpoint computation through strategy iteration. In *"European Symposium on Programming (ESOP)"*, 2007.
9. Hugo Gimbert and Wieslaw Zielonka. When can you play positionally? In Jirí Fiala, Václav Koubek, and Jan Kratochvíl, editors, *MFCS*, volume 3153 of *Lecture Notes in Computer Science*, pages 686–697. Springer, 2004.
10. Hugo Gimbert and Wieslaw Zielonka. Games where you can play optimally without any memory. In Martín Abadi and Luca de Alfaro, editors, *CONCUR*, volume 3653 of *Lecture Notes in Computer Science*, pages 428–442. Springer, 2005.
11. A.J. Hoffman and R.M. Karp. On Nonterminating Stochastic Games. *Management Sci.*, 12:359–370, 1966.
12. R. Howard. *Dynamic Programming and Markov Processes*. Wiley, New York, 1960.
13. Marcin Jurdziński. Small Progress Measures for Solving Parity Games. In *17th Ann. Symp. on Theoretical Aspects of Computer Science (STACS)*, volume 1770 of *LNCS*, pages 290–301. Springer Verlag, 2000.
14. Henrik Jrklund, Olle Nilsson, Ola Svensson, and Sergei Vorobyov. The Controlled Linear Programming Problem. Technical report, DIMACS, 2004.
15. Anuj Puri. *Theory of Hybrid and Discrete Systems*. PhD thesis, University of California, Berkeley, 1995.
16. Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, New York, 1994.
17. Helmut Seidl. A Modal μ Calculus for Durational Transition Systems. In *IEEE Conf. on Logic in Computer Science (LICS)*, pages 128–137, 1996.
18. Sharon Shoham and Orna Grumberg. Multi-valued model checking games. In Doron Peled and Yih-Kuen Tsay, editors, *ATVA*, volume 3707 of *Lecture Notes in Computer Science*, pages 354–369. Springer, 2005.
19. Jens Vöge and Marcin Jurdzinski. A Discrete Strategy Improvement Algorithm for Solving Parity Games. In *Computer Aided Verification, 12th Int. Conf. (CAV)*, pages 202–215. LNCS 1855, Springer, 2000.