Faster Solutions of Rabin and Streett Games*

Nir Piterman†
EPFL - I&C - MTC
Lausanne, Switzerland

Amir Pnueli
Weizmann Institute of Science
Rehovot, Israel

Abstract

In this paper we improve the complexity of solving Rabin and Streett games to approximately the square root of previous bounds. We introduce direct Rabin and Streett ranking that are a sound and complete way to characterize the winning sets in the respective games. By computing directly and explicitly the ranking we can solve such games in time $O(mn^{k+1}kk!)$ and space $O(nk)$ for Rabin and $O(nkk!)$ for Streett where n is the number of states, m the number of transitions, and k the number of pairs in the winning condition. In order to prove completeness of the ranking method we give a recursive fixpoint characterization of the winning regions in these games. We then show that by keeping intermediate values during the fixpoint evaluation, we can solve such games symbolically in time $O(n^{k+1}k!)$ and space $O(n^{k+1}k!)$. These results improve on the current bounds of $O(mn^{2k}k!)$ time in the case of direct (symbolic) solution or $O(m(nk^2k!)^k)$ in the case of reduction to parity games.

1 Introduction

One of the most ambitious and challenging problems in reactive system construction is the automatic synthesis of programs and (digital) designs from logical specifications. First identified as Church's problem [4], several methods have been proposed for its solution (cf. [2, 23]). The two prevalent approaches to solving the synthesis problem are by reducing it to the emptiness problem of tree automata, and viewing it as the solution of a two-person game. These two problems are essentially equivalent with efficient reductions between them [29].

A *two-player game* is a finite or infinite directed graph where the vertices are partitioned between the two players. A *play* proceeds by moving a token between the vertices of the graph. If the token is found on a vertex of player 0, she chooses an outgoing edge and moves the token along that edge. If the token is found on a vertex of player 1, she gets to choose the outgoing edge. The result is an infinite sequence of vertices. In order to determine the winner in a play we consider the *infinity set*, the set of states occurring infinitely often in the play. Then, there are several methods to define acceptance conditions that determine which infinity sets are winning for which player.

Two of the most natural such acceptance conditions are *Rabin* [22] and *Streett* [25]. Both conditions are defined using a set of pairs of subsets of the vertices of the graph. In order to win the Rabin condition over $\{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ the infinity set has to intersect G_i and not intersect R_i for some i. The Streett winning condition is the dual of the Rabin condition. In order to win the Streett condition over $\{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ the infinity set has to either be disjoint from G_i or to intersect R_i for every i. Both Rabin and Streett acceptance conditions are as general as every other ω-*regular* acceptance condition. That is, if the winning condition is defined using some automaton over infinite words (cf. [26]) or as the set of possible infinity sets (*Muller* condition) there is a way to augment the game with a *deterministic monitor* such that the winning condition over the states of the monitor is either Rabin or Streett. Another general acceptance condition is the *parity* acceptance condition [9]. In the parity condition , every vertex has a priority and a play is won if the minimal priority visited infinitely often is even. We mention parity games because our algorithms are derived from similar algorithms that solve parity games.

Rabin conditions arise naturally when the winning condition is supplied in the form of a nondeterminitic Büchi automaton over infinite words. In such a case, the standard approach to solving the game is by converting the nondeterminitic Büchi automaton to a deterministic Rabin automaton [24]. A solution to the Rabin game is then used to solve the original game.

Streett conditions arise naturally when considering synthesis of controllers from temporal logic specifications. In

*This research was supported in part by the Israel Science Foundation (grant no.106/02-1), European community project Prosyd, and the John von-Neumann Minerva center for Verification of Reactive Systems.

†Part of this research was performed while this author was visiting the Weizmann Institute.

many such cases, the controller has to supply *strong fairness*, that is, if some transition / resource is enabled / requested infinitely often it should be taken / granted infinitely often. These kind of requirements translate naturally to Streett conditions.

In [20] we presented a framework for synthesizing a design from a temporal logic specification by converting it into a two-player game, where the synthesized design plays against an adversary environment, striving to maintain the temporal specification. In that paper, we assumed that both the environment and the design are only constrained by *justice* (weak fairness) requirements. As a result of this restricting hypothesis, the resulting games were generalized Street games with $k = 1$. A strong motivation for the research reported in this paper is to remove this fairness restriction and allow *compassion* (strong fairness) both in the environment and the synthesized design. This can give rise to Street games with arbitrary k.

Consider for example the following specification of an arbiter. The arbiter, controls the grant signals for n clients. Each client, has a request signal r_i which it may raise at will. Once raised, the agent may withdraw the request but only after at least one cycle. The controller has to allocate grants (permission to access a shared resource) among the clients, so that no two clients may access the resource at the same time (*mutual exclusion*) and so that every client that requests the resource infinitely often is granted the resource infinitely often. The natural translation of this scenario into a game results in a Streett game with one strong fairness requirement for every client.

Rabin and Streett games are known to be NP-complete and co-NP-complete respectively [8]. Emerson and Jutla [8] and independently Pnueli and Rosner [21] proposed algorithms that solve Rabin and Streett games in time $O((nk)^{3k})$ where n is the number of vertices and k the number of pairs. This was later improved by Kupferman and Vardi to $O(mn^{2k}k!)$ where m is the number of edges [17]. Recently, a different solution with the same complexity was given by Horn [13]. It is also possible to solve Rabin and Streett games by reducing them to parity games [9]. This reduction is by adding a deterministic monitor with $k^2k!$ states. The resulting parity game has $nk^2k!$ states and $2k$ priorities. Using the best current solution to parity game [14], we can solve Rabin and Streett games in time $O((nk^2k!)^k)$ (enumerative algorithm).

As Rabin and Streett conditions are duals, it is enough to reason about one of them in order to *decide* the winner in a game. A player is winning according to the Streett condition iff the other player is losing according to the Rabin condition and vice versa. In order to *synthesize* programs it is not sufficient to know who is the winner; we also need the *winning strategy*. That is, what is the sequence of moves that the winning player has to perform in order to ensure her win. In order to produce the winning strategy we have to reason separately about Rabin and Streett games. This way, we can produce the winning strategy for the player that interests us (be she Rabin or Streett). It is well known that winning strategies in Rabin games are *memoryless*, i.e., depend only on the current position in the game [6]. On the other hand, winning strategies in Streett games may require exponential memory [5, 13]. It follows, that the way to produce the winning strategy may be very different.

Solutions for parity games passed also a long line of improvements. For many years, the best solution to parity games had been the symbolic fixpoint evaluation algorithm of Emerson and Lei [10, 9]. The complexity of solving a parity game using this approach is mn^k where k is the number of priorities. One major improvement of the classical algorithm has been the observation of Long et al. that by saving intermediate values of the fixpoint computation the run time can be improved to the square root, i.e., $O(n^{\frac{k}{2}})$ [19]. Long et al. show that by storing intermediate values of the fixpoint computation they can start fixpoint evaluations from better approximations. Unfortunately, the space complexity of this algorithm matches its time complexity.

Jurdziński matched the smaller upper bound while reducing space complexity to linear [14]. His algorithm computes the winning region in a parity game by computing ranks for each vertex. Every vertex with a finite rank is winning and all the rest are losing. The direct rank computation can be accomplished in time $O(mn^{\frac{k}{2}})$. A disadvantage of this approach is that it cannot be applied symbolically. Thus, forcing enumerative approach of the vertices of the game.

Here we generalize these two approaches to Rabin and Streett games. We give an enumerative algorithm that solves Rabin and Streett games in time $O(mn^{k+1}kk!)$ and $O(mn^k kk!)$ respectively and space $O(nk)$ and $O(nkk!)$ respectively. We give a symbolic algorithm that solves Rabin and Streett games in time $O(n^{k+1}k!)$ and space $O(n^{k+1}k!)$.

We introduce Rabin and Streett ranking which resemble Jurdziński's ranking in that every winning state has a finite rank and the ranking induces a winning strategy. The direct computation of these ranks requires the square root of the time of previous algorithms. Recall that in the worst case a strategy to win a Streett game may require a memory of size $k!$ [5, 13]. Thus, it seems that the memory consumption of the Streett algorithm is close to optimal.

In order to prove completeness of the ranking method we introduce recursive fixpoint algorithms that compute the winning regions in Rabin and Streett games. These algorithms match the best previous upper bounds of $O(mn^{2k}k!)$ time and resemble the fixpoint characterization of parity games [9].

We then combine the fixpoint characterization of the winning regions and Long et al.'s method of fixpoint acceleration [19]. We show that by storing intermediate values of the fixpoints in our algorithm we can accelerate the fixpoint computation by starting the computation of fixpoints from better approximations. The result is a symbolic algorithm that matches the time of the enumerative algorithm.

From our algorithms it follows that Rabin and Streett games are in fact parity games with different orders on the pairs. This has been implicit in the conversion of Rabin and Streett games to parity games, as well as in the solution of Kupferman and Vardi for Rabin games [17]. We are the first to take advantage of this connection to improve the run time of the algorithms for Rabin and Streett games almost to a factor of $k!$. We conjecture that similar generalizations can be applied to other algorithms that solve parity games [27, 1, 15].

2 Preliminaries

2.1 Linear Temporal Logic

We assume some set of Boolean variables (propositions) P. LTL formulas are constructed as follows.

$$\varphi ::= p \in P \mid \neg\varphi \mid \varphi \vee \varphi \mid \bigcirc \varphi \mid \varphi U \varphi$$

As usual we denote $\neg(\neg\varphi \vee \neg\psi)$ by $\varphi \wedge \psi$, $\mathrm{T} U \varphi$ by $\diamondsuit \varphi$ and $\neg \diamondsuit \neg\varphi$ by $\square \varphi$. For a proposition p we denote $\neg p$ by $\overline{p}$.

A *model* (alternatively, *word*) w for a formula φ is an infinite sequence of truth assignments to propositions. Namely, a word in $(2^P)^\omega$ is a model. We denote by w_i the set of propositions that are true in location i, that is $w = w_0 \cdot w_1 \cdots$. We present an inductive definition of when a formula holds in model w at time i.

- For $p \in P$ we have $w, i \models p$ iff $w_i(p) = 1$.
- $w, i \models \neg\varphi$ iff $w, i \not\models \varphi$
- $w, i \models \varphi \vee \psi$ iff $w, i \models \varphi$ or $w, i \models \psi$
- $w, i \models \bigcirc \varphi$ iff $w, i+1 \models \varphi$
- $w, i \models \varphi U \psi$ iff there exists $k \geq i$ such that $w, k \models \psi$ and forall $i \leq j < k$ we have $w, j \models \varphi$

For a formula φ and a position $j \geq 0$ such that $w, j \models \varphi$, we say that φ *holds at position* j of w. If $w, 0 \models \varphi$ we say that φ *holds* on w and denote it by $w \models \varphi$. We denote by $L(\varphi)$ the set of models that satisfy φ.

2.2 Games

A *game* is a tuple $G = \langle V, E, W \rangle$ where V is the set of states of the game, V is partitioned to V_0 and V_1 the sets of states of player 0 and player 1 respectively, $E \subseteq V \times V$ is the transition relation, and $W \subseteq V^\omega$ is the winning condition of player 0. We assume that for every $v \in V$ there exists some state $v' \in V$ such that $(v, v') \in E$.

A *play* in G is a maximal (hence infinite) sequence of locations $p = v_0 v_1 \cdots$ such that forall $i \geq 0$ we have $(v_i, v_{i+1}) \in E$. For a play p we define $inf(p)$ to be the set of states occurring infinitely often in p. Formally, $inf(p) = \{v \mid v = v_i \text{ for infinitely many } i\text{s}\}$. A play p is winning for player 0 if $p \in W$. Otherwise, player 1 wins.

A strategy for player 0 is a partial function $f : V^* \times V_0 \rightarrow V$ such that whenever $f(pv)$ is defined $(v, f(pv)) \in E$. We say that a play $p = v_0 v_1 \cdots$ is *f-conform* if whenever $v_i \in V_0$ we have $v_{i+1} = f(v_0 \cdots v_i)$. The strategy f is *winning from* v if every f-conform play that starts in v is winning for player 0. We say that *player 0 wins* from v if she has a winning strategy. The *winning region* of player 0, is the set of states from which player 0 wins. We denote the winning region of player 0 by W_0. A strategy, winning strategy, win, and winning region are defined dually for player 1. We *solve* a game by computing the winning regions W_0 and W_1. For the kind of games handled in this paper W_0 and W_1 form a partition of V [12].

In this paper we solve Rabin and Streett games. Both Rabin and Streett conditions are defined by a set of pairs of subsets of states. Formally, a Rabin condition is $\alpha = \{\langle G_1, R_1 \rangle, \ldots, \langle G_k, R_k \rangle\}$ where forall i we have G_i and R_i are subsets of V. The Rabin condition α defines the set W of infinite sequences $p \in V^\omega$ such that for some i we have $inf(p)$ intersects G_i and $inf(p)$ does not intersect R_i. A Streett condition is $\alpha = \{\langle G_1, R_1 \rangle, \ldots, \langle G_k, R_k \rangle\}$. The Streett condition α defines the set W of infinite sequences $p \in V^\omega$ such that forall i we have $inf(p)$ intersects G_i implies $inf(p)$ intersects R_i. The Streett condition is the dual of the Rabin condition; when a play is winning according to the Rabin condition it is losing according to the Streett condition and vice versa. It follows that when the winning condition for player 0 is the Rabin condition α then the Streett condition α is the winning condition for player 1. In order to partition the set of states to the winning regions it is enough to consider one of the two conditions. For example, we compute the winning region of player 0 according to the Rabin condition and its complement is the winning region for player 1 according to the Streett condition. However, when we are interested also in the winning strategy, we may be required to solve separately the Rabin and the Streett winning conditions according to the winning strategy we wish to construct. We abuse notation and write $G = \langle V, E, \alpha \rangle$ for a Rabin or Streett condition α.

For the proofs we need also winning conditions defined by general LTL formulas. In order to define the winning condition we assume that the game is equipped with a set of propositions $\mathcal{V}$ and a labeling $L : V \rightarrow 2^{\mathcal{V}}$ that labels every state v with the set of propositions that are true in it. We extend L to finite and infinite sequences of states in V and to sets of sequences of states in V in the natural way. When the winning condition for player 0 is φ then

$W = \{p \mid L(p) \in L(V^\omega) \cap L(\varphi)\}$. For example, in order to define the Rabin condition $\alpha = \{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ we treat the subsets G_i and R_i as propositions that are true for the states included in them. The Rabin condition α is then equivalent to the following LTL condition.

$$\bigvee_{i=1}^{k} (\Diamond \Box \overline{R}_i \wedge \Box \Diamond G_i)$$

2.3 μ-calculus over Game Structures

We define μ-calculus [16] over game structures. Consider a game $G = \langle V, E, \alpha\rangle$ where V is the disjoint union of V_0 and V_1 the states of player 0 and player 1, respectively. For every proposition p the formula p is an *atomic formula*. Let $Var = \{X, Y, \ldots\}$ be a set of *relational variables*. The μ-calculus formulas are constructed as follows.

$$\varphi ::= p \mid \neg p \mid X \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \Diamond\!\!\!\!\bigcirc\, \varphi \mid \Box\!\!\!\!\bigcirc\, \varphi \mid \mu X \varphi \mid \nu X \varphi$$

A formula φ is interpreted as the set of states in V in which φ is true. We write such set of states as $[[\varphi]]^e_G$ where G is the game and $e : Var \to 2^V$ is an *environment*. The environment assigns to each relational variable a subset of V. We denote by $e[X \leftarrow V']$ the environment such that $e[X \leftarrow V'](X) = V'$ and $e[X \leftarrow V'](Y) = e(Y)$ for $Y \neq X$. The set $[[\varphi]]^e_G$ is defined inductively as follows[1].

- $[[p]]^e_G = \{s \in V \mid s \models p\}$
- $[[\neg p]]^e_G = \{s \in V \mid s \not\models p\}$
- $[[X]]^e_G = e(X)$
- $[[\varphi \vee \psi]]^e_G = [[\varphi]]^e_G \cup [[\psi]]^e_G$.
- $[[\varphi \wedge \psi]]^e_G = [[\varphi]]^e_G \cap [[\psi]]^e_G$.
- $[[\Diamond\!\!\!\!\bigcirc\, \varphi]]^e_G =$
  $\{v \in V_0 \mid \exists v'$ s.t. $(v, v') \in E$ and $v' \in [[\varphi]]^e_G\} \quad \cup$
  $\{v \in V_1 \mid \forall v'$ s.t. $(v, v') \in E$ we have $v' \in [[\varphi]]^e_G\}$
  A state v is included in $[[\Diamond\!\!\!\!\bigcirc\, \varphi]]^e_G$ if player 0 can force the play to reach a state in $[[\varphi]]^e_G$. That is, either v is a state of player 0 and has some successor in $[[\varphi]]^e_G$ or v is a state of player 1 and all its successors are in $[[\varphi]]^e_G$.
- $[[\Box\!\!\!\!\bigcirc\, \varphi]]^e_G =$
  $\{v \in V_1 \mid \exists v'$ s.t. $(v, v') \in E$ and $v' \in [[\varphi]]^e_G\} \quad \cup$
  $\{v \in V_0 \mid \forall v'$ s.t. $(v, v') \in E$ we have $v' \in [[\varphi]]^e_G\}$
  A state v is included in $[[\Box\!\!\!\!\bigcirc\, \varphi]]^e_G$ if player 1 can force the play to reach a state in $[[\varphi]]^e_G$. That is, either v is a state of player 1 and has some successor in $[[\varphi]]^e_G$ or v is a state of player 0 and all its successors are in $[[\varphi]]^e_G$.
- $[[\mu X \varphi]]^e_G = \cup_i S_i$ where $S_0 = \emptyset$ and $S_{i+1} = [[\varphi]]^{e[X \leftarrow S_i]}_G$.
- $[[\nu X \varphi]]^e_G = \cap_i S_i$ where $S_0 = V$ and $S_{i+1} = [[\varphi]]^{e[X \leftarrow S_i]}_G$

[1] Only for games with a finite number of states.

When all the variables in φ are bound by either μ or ν the initial environment is not important and we simply write $[[\varphi]]_G$. In case that G is clear from the context we simply write $[[\varphi]]$.

Consider for example a game $G = \langle V, E, W\rangle$ and the formula $\varphi = \nu X(p \wedge \Diamond\!\!\!\!\bigcirc\, X)$. A state $v \in V$ is in $[[\nu X(p \wedge \Diamond\!\!\!\!\bigcirc\, X)]]$ if player 0 can force the game to remain in p states forever. Indeed player 0 can force the game to another state in $[[\nu X(p \wedge \Diamond\!\!\!\!\bigcirc\, X)]]$ and so on ad-infinitum.

The formula $\psi = \mu X(\neg p \vee \Box\!\!\!\!\bigcirc\, X)$ characterizes the set of states from which player 1 can force a visit to a $\neg p$ state. Indeed, player 1 can force the game in a finite number of steps to the set $[[\neg p]]$.

We freely use μ-calculus formulas with complex operators that compute sets of states. In such a case we simply use the set returned by the operator in the inductive definition of the μ-calculus. For a full exposition of μ-calculus we refer the reader to [7]. We often abuse notations and write a μ-calculus formula φ instead of the set $[[\varphi]]$.

3 Rabin and Streett Ranking

In this section we show how to define Rabin and Streett ranking. We show that our ranking induces a winning strategy for player 0. We show that our ranking is defined on the winning region. Intuitively, the ranking measure the distance towards achieving small milestones during a play. By reducing the distance to these milestones we get to them, which eventually leads us to winning the game.

3.1 Rabin Ranking

Consider a game $G = \langle V, E, \alpha\rangle$ where $\alpha = \{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ is a Rabin winning condition. Player 0 wins an infinite play p if there exists $\langle G_i, R_i\rangle \in \alpha$ such that $inf(p) \cap G_i \neq \emptyset$ and $inf(p) \cap R_i = \emptyset$. We now define formally the range of the ranking function and the ranking function itself.

Let $\Pi(k)$ denote the set of permutations over $[1..k]$. Given a permutation $\pi = j_1 j_2 \cdots j_k \in \Pi(k)$ we denote j_i by π_i. The *Rabin domain* for α over V is $D_R(\alpha, V) = \{i_0 j_1 i_1 j_2 \cdots j_k i_k \mid i_0 \cdots i_k \in [0..n]^{k+1}$ and $j_1 \cdots j_k \in \Pi(k)\} \cup \{\infty\}$. That is, the domain contains the interleaving of a $k+1$ tuple of integers with a permutation over $[1..k]$. Every integer is bounded by n. For simplicity of notations we write D_R and Π instead of $D_R(\alpha, V)$ and $\Pi(k)$. Given $d = i_0 j_1 \cdots j_k i_k \in D_R$ we denote by $\pi(d)$ the permutation $j_1 \cdots j_k$ and by $m(d)$ the tuple $i_0 \cdots i_k \in [0..n]^k$. We order D_R according to the lexicographic ordering with ∞ as maximal element.

A *Rabin ranking* over V is $r : V \to D_R$. Intuitively, the ranking $i_0 j_1 \cdots j_k i_k$ fixes an order $j_1 \cdots j_k$ on the Rabin pairs. This is the order of importance between the pairs.

It means that it is most important to visit G_{j_1} while avoiding R_{j_1}. We are also happy if we avoid R_{j_1} and R_{j_2} and visit G_{j_2} infinitely often and so on. A visit to R_{j_l} is allowed only by changing the importance order of the pairs that are less important than j_l (and j_l itself). We allow the order to change only to lower orders (according to the lexicographic ordering on permutations). This means that R_{j_l} can be visited only finitely often. The value i_l in the sequence $i_0 \cdots i_k$ measures the worst possible number of steps until a visit to G_{j_l} (while avoiding $R_{j_{l'}}$ forall $l' \leq l$). Whenever we visit G_{j_l} we are so happy that we allow to change the order of the less important pairs and to increase the distance to Gs for less important pairs. Finally, i_0 is intuitively the number of times that R_{j_1} may be visited (forcing a change to a lower permutation). Formally, we have the following.

Given a node $v \in V$ and a Rabin ranking r we denote by $best(v)$ the rank of the minimal successor of v in case that $v \in V_0$ or the rank of the maximal successor of V in case that $v \in V_1$. Formally,

$$best(v) = \begin{cases} min_{(v,w)\in E}(r(w)) & v \in V_0 \\ max_{(v,w)\in E}(r(w)) & v \in V_1 \end{cases}$$

We say that a Rabin ranking is *good* if for every state v such that $r(v) \neq \infty$ we have $best(v)$ is *better* than $r(v)$. Let $r(v) = i_0 j_1 i_1 \cdots i_k$ and and $best(v) = i'_0 j'_1 i'_1 \cdots i'_k$. We say that $best(v)$ is *better* than $r(v)$ if $i_0 > i'_0$ or $i_0 = i'_0$ and $best(v)$ is *better*$_1$ than $r(v)$. We say that $best(v)$ is *better*$_l$ than $r(v)$ if one of the following holds.

- $j_l > j'_l$.
- $j_l = j'_l$, $v \models \overline{R}_{j_l}$, and $i_l > i'_l$.
- $j_l = j'_l$, $v \models \overline{R}_{j_l}$, and $v \models G_{j_l}$.
- $j_l = j'_l$, $v \models \overline{R}_{j_l}$, $i_l = i'_l$, and $best(v)$ is better$_{l+1}$ than $r(v)$.

If one of the first three conditions holds we say that $best(v)$ is *strictly better*$_l$ than $r(v)$. It is simple to see that if $v \in V_1$ and $best(v)$ is better than $r(v)$ then for every node w such that $(v, w) \in E$ we have $r(w)$ is better than $r(v)$. This follows from $r(w)$ being at most $best(v)$.

We show that Rabin ranking is sound and complete. We show soundness by proving that the strategy of choosing the minimal possible successor is winning for player 0. Consider a play where player 0 uses this strategy. It follows that the sequence of ranks gets better and better (i.e., the rank of every state is better than that of its predecessor). The only way to create an infinite sequence of ranks that get better is by allowing the suffix of the rank to increase (i.e., leave the prefix $i_0 \cdots j_l$ fixed and increase $i_l j_{l+1} \cdots j_k i_k$). By the definition of better, the only way to increase the suffix of the rank is for some l to have that the rank is strictly better$_l$. There is some minimal l for which the ranks get strictly better$_l$ infinitely often. Consider the point in the play from which the ranks are always better$_l$ and infinitely often strictly better$_l$. In order to visit R_{j_l} the rank has to be strictly better$_{l'}$ for some $l' < l$ and this is impossible. Thus, R_{j_l} is never visited beyond this point. In order to allow infinitely many strictly better$_l$, it has to be the case that G_{j_l} is visited infinitely often. Formally, we have the following.

Claim 1 *Given a good Rabin ranking r, player 0 wins the Rabin game from every state v such that $r(v) \neq \infty$.*

Proof: Consider the following strategy. From a state $v \in V_1$ choose the successor w such that $r(w)$ is minimal. We show that this strategy is winning.

Consider an infinite play $v_0 v_1 \cdots$ that conforms to this strategy. Let $r_0 r_1 \cdots$ denote the sequence of ranks such that $r_m = r(v_m)$ and $r_m = i_0^m j_1^m i_1^m \cdots j_k^m i_k^m$. From the definition of good ranking it follows that it is always the case that r_{m+1} is better than r_m. Let l be the minimal value such that there exist infinitely many m such that r_{m+1} is strictly better$_l$ than r_m. There exists m' such that forall $m > m'$ and forall $l' < l$ we have r_{m+1} is not strictly better$_{l'}$ from r_m. So for all $l' < l$ and forall $m > m'$ we have $j_{l'}^m = j_{l'}^{m+1}$, $v_m \models \overline{R}_{j_{l'}^m}$, and $i_{l'}^m = i_{l'}^{m+1}$. Similarly, there exists $u \in [1..k]$ and $m'' > m'$ such that forall $m > m'$ we have $j_l^m = u$. Consider the pair $\langle G_u, R_u \rangle$ and the suffix of the play starting from m''. For every $m > m''$ we have r_{m+1} is better than r_m, hence $v_m \models \overline{R}_u$. Furthermore, whenever r_{m+1} is strictly better$_u$ than r_m then either $i_u^m > i_u^{m+1}$ or $v_m \models G_u$. We conclude that the play is winning according to the pair $\langle G_u, R_u \rangle$. □

We show that the algorithm in Fig. 5 induces a good Rabin ranking. Thus, proving completeness of the Rabin ranking method.

Claim 2 *For every Rabin game there exists a good Rabin ranking such that for every state v winning for player 0 according to the Rabin winning condition we have $r(v) \neq \infty$.*

Proof: Denote by W the set of states returned by the algorithm in Fig. 5. We show how to define a good Rabin ranking on the states in W. In order to define the ranking we analyze the way the computation advances. The analysis is similar to the analysis of the fixpoint computation in [28]. Formally, we have the following.

In every stage of the computation we record the status of the call stack. According to the contents of the call stack we define sets of states whose union includes all the states in W. We then use these sets to give ranks to the states in W. First, let us add a counter to the least fixpoints. We assume that with the minimal fixpoint there is a counter i. This counter is initialized to zero in the first visit to line 1 in the function `main_Rabin` and increased by one in every subsequent visit. Similarly, the counter is initialized to 0 in the first visit to line 6 in the function `Rabin` and increased by one in every subsequent visit.

Consider the state of the call stack when the computation reaches line 1 in function `main_Rabin`. We use the counter i to set Z_j to the value of Z in the iteration where i is incremented to j. It follows that in the first iteration when i is initialized to 0 we have $Z = \emptyset$ and we set $Z_0 = \emptyset$. Furthermore, $Z_{i+1} = \texttt{Rabin(Set,true,cpred(}Z_i\texttt{))}$.

We monitor the call stack if every copy of `Rabin` on the call stack is found in the last iteration of the maximal fixpoint. That is, the value of Y (in each copy) is already the value computed by the next iteration. In what follows, every configuration of the call stack is assumed to be in such a state. Consider a configuration of the call stack where the active copy of `Rabin` is in line 6. Let us denote the number of copies of the function `Rabin` on the call stack by l. Let $j_1 \cdots j_l$ be the pairs of the Rabin condition handled by these copies of `Rabin` and let $i_0 \cdots i_l$ be the values of the counter i (where i_0 is the counter in the function `main_Rabin`). We set $X^{i_0 \cdots i_l}_{j_1 \cdots j_l}$ to be the value of X in the active copy of `Rabin` in this state of the call stack. Again, whenever i_l is 0 we have $X^{i_0 \cdots i_l}_{j_1 \cdots j_l}$ is the empty set.

Consider a tuple $i_0 \cdots i_l$ and a prefix of a permutation $j_1 \cdots j_l$. From the structure of the fixpoint it follows that $X^{i_0 \cdots (i_l+1)}_{j_1 \cdots j_l}$ is exactly the union of $X^{i_0 \cdots i_l i}_{j_1 \cdots j_l j}$ for every value of $j \notin \{j_1, \ldots, j_l\}$ and i.

For every state $v \in W$ there exists $d \in D_R$ such that $v \in X^{m(d)}_{\pi(d)}$ and d is minimal according to the ordering on D_R. We set $r(v)$ to be this minimal value d. For all states $v \notin W$ we set $r(v) = \infty$. We show that the resulting ranking is a good Rabin ranking.

Consider a state $v \in W$. Let $r(v) = i_0 j_1 i_1 \cdots j_k i_k$. Consider the call stack of the computation at the point where $X^{i_0 \cdots i_k}_{j_1 \cdots j_k}$ is computed. Let $Y_{j_1 j_2 \cdots j_l}$ denote the value of Y in the lth copy of `Rabin` on the call stack (counting from the bottom of the stack). Notice, that we do not have to annotate Y by $i_0 i_1 \cdots i_{l-1}$ as we are considering only the specific rank $r(v)$. From the structure of the fixpoint it follows that $Y_{j_1 \cdots j_l}$ is exactly the union of $X^{i_0 \cdots i_l i}_{j_1 \cdots j_l j}$ for all possible values of i and $j \notin \{j_1 \cdots j_l\}$.

By flattening the function calls of the recursive algorithm we get that $X^{i_o \cdots i_k}_{j_1 \cdots j_k}$ is equivalent to the expression in Fig. 1. Consider some $v \in X^{i_0 \cdots i_k}_{j_1 \cdots j_k}$. If v is in the first disjunct $\diamondsuit Z_{i_0 - 1}$ then *best*(v) is better than $r(v)$ (without checking better$_1$).

If v is in the $a+1$th disjunct

$$\begin{array}{l}\left((\bigwedge_{l=1}^{a} \overline{R}_{j_l}) \wedge \diamondsuit X^{i_0 \cdots (i_a - 1)}_{j_1 \cdots j_a} \right) \quad \vee \\ \left((\bigwedge_{l=1}^{a} \overline{R}_{j_l}) \wedge \diamondsuit Y_{j_1 \cdots j_a} \right)\end{array}$$

and v is not in all the disjuncts below it then *best*(v) is better$_{a'}$ from $r(v)$ for all $a' < a$ (but not strictly better$_{a'}$). This follows from $i_0 \cdots i'_a$ being equivalent and from $\overline{R}_1 \wedge \cdots \wedge \overline{R}_a$ holding in v. It is also the case that *best*(v) is strictly better$_a$ than $r(v)$. If $v \in (\bigwedge_{l=1}^{a} \overline{R}_{j_l}) \wedge \diamondsuit X^{i_0 \cdots i_a}_{j_1 \cdots j_a}$ the ath coordinate of the ranking decreases. If $v \in (\bigwedge_{l=1}^{a} \overline{R}_{j_l}) \wedge G_{j_l} \diamondsuit Y_{j_1 \cdots j_a}$ then v is a G_{j_l} state.

We conclude that from v player 0 can control the game so that the successor of v is better than v. □

Theorem 3 *Player 0 wins the Rabin game from v iff there exists a good Rabin ranking such that $r(v) \neq \infty$.*

3.2 Streett Ranking

Consider a game $G = \langle V, E, \alpha \rangle$ where $\alpha = \{\langle G_1, R_1 \rangle, \ldots, \langle G_k, R_k \rangle\}$ is a Streett winning condition. Player 0 wins an infinite play p if forall i we have $inf(p) \cap G_i \neq \emptyset$ implies $inf(p) \cap R_i \neq \emptyset$. We now define formally the range of the ranking function and the ranking function itself.

The *Streett domain* for α over V is $[0..n]^k \cup \{\infty\}$, denoted by $D_S(\alpha, V)$. We order $D_S(\alpha, V)$ according to the lexicographic order with ∞ as maximal element. Given $m \in D_S(\alpha, V)$ we denote by m_l the lth entry in m. Consider the set $\Pi(k)$. Let $\pi = j_1 \cdots j_k \in \Pi(k)$ be some permutation. We define what does it mean to increase the lth entry in π. We increase the lth entry by leaving the first $l-1$ entries unchanged. For the lth entry we choose the next available value among the rest of the entries. If the lth entry is already the maximal among these entries then we go back to the minimal. The rest are ordered in increasing order. Let $\pi = j_1 \cdots j_k$. We set $inc_l(\pi)$ to be the permutation $j_1 \cdots j_{l-1} j'_l \cdots j'_k$ such that if $j_l = max(j_l, \ldots j_k)$ then $j'_l = min(j_l, \cdots, j_k)$ and if $j_l < max(j_l, \ldots, j_k)$ then j'_l is set to the minimal value in $j_l, \ldots, j_k$ such that $j'_l > j_l$. Then, we order $\{j_l, \ldots, j_k\} - \{j'_l\}$ in increasing order and this completes the permutation. For example, $inc_k(\pi)$ is π, $inc_1(123)$ is 213, and $inc_2(123)$ is 132. For simplicity of notations, we write D_S and Π instead of $D_S(\alpha, V)$ and $\Pi(k)$.

A *Streett ranking* over V is $r : V \times \Pi \to D_S$. That is, with every state $v \in V$ and every permutation $\pi \in \Pi$ we associate a rank in D_S. Intuitively, the ranking $r(v, \pi) = i_1 \cdots i_k$ is a rank according to the order π on the pairs. As before, it is most important to visit R_{j_1}. We are also happy if we avoid G_{j_1} and visit R_{j_2} and so on. Intuitively, i_l counts how many visits to G_{j_l} are possible until a visit to R_{j_l}. In particular, either G_{j_l} is visited finitely often, or after every visit to G_{j_l} there is a visit to R_{j_l}. Whenever we visit R_{j_l} we switch to pursue a visit to $R_{j_{l'}}$ for one of the next 'less important' pairs. We do this by replacing the permutation π by a permutation π' that agrees with π on the $l-1$ first entries. Thus, we continue to avoid $G_{j_{l''}}$ for $l'' < l$ and visit (infinitely often) $R_{j_{l'}}$ for $l' \geq l$. Formally, we have the following.

For every state v and permutation π, we denote by *best*(v, π) the rank of the minimal successor of v in case that $v \in V_0$ or the rank of the maximal successor of V in

$$
\begin{array}{ll}
X^{i_0 \cdots i_k}_{j_1 \cdots j_k} = & \\
\bigotimes Z_{i_0 - 1} & \vee \\
\left(\overline{R}_{j_1} \wedge \bigotimes X^{i_0 (i_1 - 1)}_{j_1}\right) \vee \left(\overline{R}_{j-1} \wedge G_{j-1} \wedge \bigotimes Y_{j_1}\right) & \vee \\
\left(\overline{R}_{j_1} \wedge \overline{R}_{j_2} \wedge \bigotimes X^{i_0 i_1 (i_2 - 1)}_{j_1 j_2}\right) \vee \left(\overline{R}_{j_1} \wedge \overline{R}_{j_2} \wedge G_{j_2} \wedge \bigotimes Y_{j_1 j_2}\right) & \vee \\
\vdots & \\
\vdots & \\
\vdots & \\
\left(\left(\bigwedge_{l=1}^{k-1} \overline{R}_{j_l}\right) \wedge \bigotimes X^{i_0 \cdots (i_{k-1} - 1)}_{j_1 \cdots j_{k-1}}\right) \vee \left(\left(\bigwedge_{l=1}^{k-1} \overline{R}_{j_l}\right) \wedge G_{j_{k-1}} \wedge \bigotimes Y_{j_1 \cdots j_{k-1}}\right) & \vee \\
\left(\left(\bigwedge_{l=1}^{k} \overline{R}_{j_l}\right) \wedge \bigotimes X^{i_0 \cdots (i_k - 1)}_{j_1 \cdots j_k}\right) \vee \left(\left(\bigwedge_{l=1}^{k} \overline{R}_{j_l}\right) \wedge G_{j_k} \wedge \bigotimes Y_{j_1 \cdots j_k}\right) &
\end{array}
$$

Figure 1. Unwinding of Recursive Algorithm.

case that $v \in V_1$. Let $\pi = j_1 \cdots j_k$, if $v \in R_{j_l}$ for some l then we consider the rank of the successors according to the permutation $inc_l(\pi)$. Formally,

$$
best(v,\pi) = \begin{cases} min_{(v,w)\in E}(r(w, inc_l(\pi))) & v \in V_0 \text{ and } v \in R_{j_l} \\ min_{(v,w)\in E}(r(w,\pi)) & v \in V_0 \text{ and } \forall l.\ v \notin R_{j_l} \\ max_{(v,w)\in E}(r(w, inc_l(\pi))) & v \in V_1 \text{ and } v \in R_{j_l} \\ max_{(v,w)\in E}(r(w,\pi)) & v \in V_1 \text{ and } \forall l.\ v \notin R_{j_l} \end{cases}
$$

We say that a Streett ranking is *good* if for every state v and $\pi \in \Pi$ such that $r(v,\pi) \neq \infty$ we have $best(v,\pi)$ is *better* than $r(v,\pi)$. Let $\pi = j_1 \cdots j_k$, $r(v,\pi) = i_1 \cdots i_k$, and $best(v,\pi) = i'_1 \cdots i'_k$. We say that $best(v,\pi)$ is *better* than $r(v,\pi)$ if it is *better*$_1$ than $r(v,\pi)$. We say that $best(v,\pi)$ is *better*$_l$ than $r(v,\pi)$ if one of the following holds.

- $i_l > i'_l$.
- $v \models R_{j_l}$ and $best(v, inc_l(\pi)) \neq \infty$.
- $i_l = i'_l$, $v \models \neg G_{j_l}$, and $best(v,\pi)$ is better$_{l+1}$ than $r(v,\pi)$.

Finally, $best(v,\pi)$ is *better*$_{k+1}$ than $r(v,\pi)$ if $best(v,\pi) \neq \infty$. It is simple to see that if $v \in V_1$ and $best(v,\pi)$ is better than $r(v,\pi)$ then for every node w such that $(v,w) \in E$ we have $r(w,\pi)$ is better than $r(v,\pi)$.

We show that Streett ranking is sound and complete. We show soundness by proving that the rank induces a winning strategy. Player 0 uses a permutation in Π_k as memory value. As long as the memory value is π, player 0 uses the ranking $r(\cdot,\pi)$ to determine her next move. While playing with memory $\pi = j_1 \cdots j_k$, player 0 tries to minimize the rank $r(\cdot,\pi)$. Whenever the set R_{j_l} is visited, player 0 chooses the least j' in $j_{l+1}, \ldots, j_k$ that is greater than j_l (if no such value exists then the minimal in $j_{l+1}, \ldots, j_k$) and changes her memory value to $j_1 \cdots j_{l-1} j', j'_{l+1} \cdots j'_k$ where $j'_{l+1} \cdots j'_k$ are the remaining pairs in increasing order. Consider a play where player 0 uses this strategy. It follows that as long as the memory does not change all parts G of pairs are not visited. One option is to eventually remain with constant memory, which implies that $G_{l'}$ forall l' are visited finitely often. Otherwise, the memory changes infinitely often. There is a point l for which the memory changes around point l infinitely often. It follows that all $G_{l'}$ for $l' < l$ are visited finitely often and all $R_{l''}$ for $l'' \geq l$ are visited infinitely often. Formally, we have the following.

Claim 4 *Given a good Streett ranking r, player 0 wins the Streett game from every state v such that for some permutation $\pi \in \Pi$ we have $r(v,\pi) \neq \infty$.*

Proof: We construct a strategy that uses as memory a permutation from Π. The initial value of this memory is a permutation π such that $r(v,\pi) \neq \infty$. We define the strategy.

From a state $v \in V_0$ with memory $\pi \in \Pi$ apply *policy*$_1$. Let $\pi = j_1 \cdots j_k$, $r(v,\pi) = i_1 \cdots i_k$, and $best(v,\pi) = i'_1 \cdots i'_k$. In order to apply *policy*$_l$ we do the first possible option of the following.

- If $i'_l < i_l$ then choose w for which $r(w,\pi) = best(v,\pi)$.
- If $v \models R_{j_l}$, update the memory to $\pi' = inc_l(\pi)$. Choose some successor w such that $r(w, inc_l(\pi)) = best(v,\pi)$.
- If $i'_l = i_l$ and $v \models \overline{R}_{j_l}$ then apply policy$_{l+1}$.

In order to apply policy $k+1$ we simply choose some successor w for which $r(w,\pi) = best(v,\pi)$. It is simple to see that if the Streett ranking is good then from a state v and permutation π such that $r(v,\pi)$ is finite it is possible to apply this strategy. We have to show that this strategy is winning.

Consider an infinite play $v_0 v_1 \cdots$ that conforms to this strategy and let $\pi_0 \pi_1 \cdots$ be the sequence of memory values that is used in the application of the strategy. Let $\pi_m = j_1^m \cdots j_k^m$. Let $r_0 r_1 \cdots$ denote the sequence of ranks such that $r_m = r(v_m, \pi_m)$ and let $r_m = i_1^m \cdots i_k^m$. We have to show that $v_0 v_1 \cdots$ is winning for player 0.

Let l be the minimal value such that there are infinitely many locations such that policy$_l$ is applied while policy$_{l+1}$ is not applied (that is, one of the first two options in policy$_l$ is chosen). There exists m' such that forall $m > m'$ it is always the case that policy$_l$ is applied (sometimes by calling policy$_{l+1}$). It follows that there exist values $j_1 \cdots j_{l-1}$ such that forall $m > m'$ we have $j_1^m \cdots j_{l-1}^m = j_1 \cdots j_{l-1}$. From the definition of good ranking and the strategy it follows that forall $m > m'$, forall $u < l$ we have $v_m \notin G_{j_u}$. Hence, all the pairs $\langle G_{j_u}, R_{j_u} \rangle$ for $u < l$ are satisfied. Consider the values $j_l \cdots j_k$. As policy$_l$ is applied infinitely often it follows that for every $u \geq l$ we have R_{j_u} is visited infinitely often. It follows that also the pairs $\langle G_u, R_u \rangle$ for $u \geq l$ are satisfied and the play is winning for player 0. □

We show that the algorithm in Fig. 6 induces a good Streett ranking. Thus, proving completeness of the Streett ranking method.

Claim 5 *For every Streett game there exists a good Streett ranking such that for every state v winning for player 1 according to the Streett winning condition there exists a permutation π such that $r(v, \pi) \neq \infty$.*

Proof: Denote by W the set of states returned by the algorithm in Fig. 6. We show how to define a good Streett ranking on the states in W. In order to define the ranking we analyze the way the computation advances. The analysis is similar to the analysis of the fixpoint computation in [28]. Formally, we have the following.

In every stage of the computation we record the status of the call stack. According to the contents of the call stack we define sets of states that include all the states in W. We then use these sets to give ranks to the states in W. First, let us add a counter to the least fixpoints. We assume that with the minimal fixpoint there is a counter i. This counter is initialized to zero in the first visit to line 5 in the function `Streett` and increased by one in every subsequent visit.

We monitor the call stack if every copy of `Streett` on the call stack is found in the last iteration of the maximal fixpoint. That is, the value of Z (in each copy) is already the value computed by the next iteration. In what follows, every configuration of the call stack is assumed to be in such a state. Consider a configuration of the call stack where the active copy of `Streett` is in line 5. Let us denote the number of copies of the function `Streett` on the call stack by l. Let $j_1 \cdots j_l$ be the pairs of the acceptance condition handled by these copies of `Streett` and let $i_1 \cdots i_l$ be the values of the counter i. We set $Y^{i_1 \cdots i_l}_{j_1 \cdots j_l}$ to be the value of Y in the active copy of `Streett` in this state of the call stack. It follows that whenever i_l is 0 we have $Y^{i_1 \cdots i_l}_{j_1 \cdots j_l}$ is the empty set.

Consider a tuple $i_0 \cdots i_l$ and a prefix of a permutation $j_1 \cdots j_l$. From the structure of the fixpoint it follows that for every value $j \notin \{j_1, \ldots, j_l\}$ we have $Y^{i_0 \cdots (i_l+1)}_{j_1 \cdots j_l}$ is exactly the union of $Y^{i_0 \cdots i_l i}_{j_1 \cdots j_l j}$ for every possible value i.

For every state $v \in W$ and every permutation $\pi \in \Pi$ such that there exists $d \in D_S$ such that $v \in Y^d_\pi$, we set $r(v, \pi)$ to be the minimal such value d. From the definition of the fixpoint for every value $v \in W$ there exists at least one such permutation $\pi \in \Pi$. For all states $v \notin W$ and forall permutations $\pi \in \Pi$ we set $r(v, \pi) = \infty$. We show that the resulting ranking is a good Streett ranking.

Consider a state $v \in W$ and some permutation π such that $r(v, \pi) < \infty$. Let $\pi = j_1 \cdots j_k$ and $r(v, \pi) = i_1 \cdots i_k$. Consider the call stack of the computation at the point where $Y^{r(v,\pi)}_\pi$ is computed. Let $Z_{j_1 \cdots j_l}$ denote the value of Z in the $l + 1$th copy of `Streett` on the call stack (counting from the bottom of the stack). Notice that we do not have to annotate $Z_{j_1 \cdots j_l}$ by $i_1 \cdots i_l$ as we are considering only the specific rank $r(v, \pi)$. Notice as well that according to this notation Z is the winning set computed by the algorithm. Let $X_{j_1 \cdots j_k}$ denote the value of X returned by the function `m_Streett`. From the structure of the fixpoint it follows that for every $j \notin \{j_1 \cdots j_l\}$ we have $Z_{j_1 \cdots j_l}$ is exactly the union of $Y^{i_1 \cdots i_l i}_{j_1 \cdots j_l j}$ for all possible values of i.

By flattening the function calls of the recursive algorithm we get that $Y^{i_1 \cdots i_k}_{j_1 \cdots j_k}$ is equivalent to the expression in Fig. 2. Consider some $v \in Y^{i_1 \cdots i_k}_{j_1 \cdots j_k}$. Let $\pi = j_1 \cdots j_k$. If v is in the first disjunct $(q_{j_1} \wedge \diamondsuit Z) \vee \diamondsuit Y^{i_1 - 1}_{j_1}$ then $best(v, \pi)$ is better$_1$ than $r(v, \pi)$. If $v \in (q_{j_1} \wedge \diamondsuit Z_{j_1})$ then v is a q_{j_1} state and as for every j we have Z is equal to $\bigcup_i Y^i_j$ it follows that $best(v, inc_1(\pi)) \neq \infty$.

If v is in the $a + 1$th disjunct

$$\begin{array}{l} \left((\bigwedge_{l=1}^{a} p_{j_l}) \wedge q_{j_{a+1}} \wedge \diamondsuit Z_{j_1 \cdots j_a} \right) \quad \vee \\ \left((\bigwedge_{l=1}^{a} p_{j_l}) \wedge \diamondsuit Y^{i_1 \cdots (i_{a+1} - 1)}_{j_1 \cdots j_{a+1}} \right) \end{array}$$

and v is not in all the disjuncts above it then $best(v, \pi)$ is better$_{a'}$ from $r(v, \pi)$ for all $a' < a + 1$. This follows from the $i_0 \cdots i'_a$ being equivalent and from $p_1 \wedge \cdots \wedge p_{a'}$ holding in v. It is also the case that $best(v, \pi)$ is better$_{a+1}$ then $r(v, \pi)$. If $v \in (\bigwedge_{l=1}^{a} p_{j_l}) \wedge q_{j_{a+1}} \wedge \diamondsuit Z_{j_1 \cdots j_a}$ then v is a $q_{j_{a+1}}$ state. As for every $j \notin \{j_1 \cdots j_a\}$ we have $Z_{j_1 \cdots j_a}$ is $\bigcup_i Y^{i_1 \cdots i_a i}_{j_1 \cdots j_a j}$ it follows that $best(v, inc_{a+1}(\pi)) \neq \infty$ and $best(v, \pi)$ is better$_{a+1}$ than $r(v, \pi)$. If $v \in (\bigwedge_{l=1}^{a} p_{j_l}) \wedge \diamondsuit Y^{i_1 \cdots (i_a - 1)}_{j_1 \cdots j_a}$ then the $a + 1$th coordinate of $best(v, \pi)$ decreases. Finally, if v is in the $k + 1$th disjunct $(\bigwedge_{l=1}^{k} p_{j_1}) \wedge \diamondsuit X_{j_1 \cdots j_k}$ then $best(v, \pi)$ is better$_{k+1}$ than $r(v, \pi)$.

We conclude that from v player 1 can control the game so that the successor of v is better than v. □

Theorem 6 *Player 1 wins the Streett game from v iff there exists a good Streett ranking and permutation π such that $r(v, \pi) \neq \infty$.*

$$
\begin{array}{ll}
Y^{i_1\cdots i_k}_{j_1\cdots j_k} = & \\
\quad (q_{j_1} \wedge \bigcirc Z) \vee \left(\bigcirc Y^{(i_1-1)}_{j_1}\right) & \vee \\
\quad (p_{j_1} \wedge q_{j_2} \wedge \bigcirc Z_{j_1}) \vee \left(p_{j_1} \wedge \bigcirc Y^{i_1\cdots(i_2-1)}_{j_1 j_2}\right) & \vee \\
\quad \vdots & \\
\quad \left(\left(\bigwedge_{l=1}^{k-2} p_{j_l}\right) \wedge q_{j_{k-1}} \wedge \bigcirc Z_{j_1\cdots j_{k-2}}\right) \vee \left(\left(\bigwedge_{l=1}^{k-2} p_{j_l}\right) \wedge \bigcirc Y^{i_1\cdots(i_{k-1}-1)}_{j_1\cdots j_{k-1}}\right) & \vee \\
\quad \left(\left(\bigwedge_{l=1}^{k-1} p_{j_l}\right) \wedge q_{j_k} \wedge \bigcirc Z_{j_1\cdots j_{k-1}}\right) \vee \left(\left(\bigwedge_{l=1}^{k-1} p_{j_l}\right) \wedge \bigcirc Y^{i_1\cdots(i_k-1)}_{j_1\cdots j_k}\right) & \vee \\
\quad \left(\left(\bigwedge_{l=1}^{k} p_{j_l}\right) \wedge \bigcirc X_{j_1\cdots j_k}\right) &
\end{array}
$$

Figure 2. Unwinding of Recursive Algorithm.

4 Computing Ranks Explicitly

So far we have established the existence of good ranking systems for Rabin and Streett games. We do not know yet how to compute such rankings. In this section we generalize Jurdziński's explicit ranking computation of parity games to Rabin and Streett ranking [14]. As in the case of parity, the minimal good ranking is a least fixpoint of a monotone operator on a complete lattice. By Knaster-Tarski theorem there exists a least good ranking and there exists a simple lifting algorithm that computes it. From previous section it follows that the least good ranking is defined on the winning region. Etessami et al. show exactly how to encode Jurdziński's algorithm to get the stated time and space bounds [11]. We extend their efficient implementation to the more general case of Rabin and Streett rankings.

Consider the set of possible Rabin rankings $r : V \rightarrow D_R$. We say that $r_1 \sqsubseteq r_2$ if for every $v \in V$ we have $r_1(v) \leq r_2(v)$. The resulting structure is a complete lattice. We use $r_1 \sqsubset r_2$ to denote $r_1 \sqsubseteq r_2$ and $r_1 \neq r_2$. We now define the lifting operator. Given a ranking $r : V \rightarrow D_R$ and a state $v \in V$ we set $prog(r, v)$ to be the least value $d \in D_R$ such that $best(v)$ is better than d. We define $lift(r, v)$ to be the following function.

$$
lift(r,v)(u) = \begin{cases} r(u) & u \neq v \\ max\{r(u), prog(r,u)\} & u = v \end{cases}
$$

The operator *lift* is monotone according to $\sqsubseteq$. Furthermore, every good Rabin ranking r is a pre-fixpoint with respect to $lift(r, v)$ for all states $v \in V$ and every pre-fixpoint with respect to $lift(r, v)$ for all states $v \in V$ is a good Rabin ranking.

Similarly, consider the set of possible Streett rankings $r : V \times \Pi \rightarrow D_S$. We say that $r_1 \sqsubseteq r_2$ if for every $v \in V$ and every $\pi \in \Pi$ we have $r_1(v, \pi) \leq r_2(v, \pi)$. The resulting structure is a complete lattice. We use $r_1 \sqsubset r_2$ to denote $r_1 \sqsubseteq r_2$ and $r_1 \neq r_2$. The Streett lifting operator is defined analogously to the above. Given a ranking $r : V \times \Pi \rightarrow D_S$, a state $v \in V$, and a permutation $\pi \in \Pi$ we set $prog(r, v, \pi)$ to be the least value $d \in D_S$ such that $best(v, \pi)$ is better than d. The ranking $lift(r, v, \pi)$ is the following ranking.

$$
lift(r,v,\pi)(u) = \begin{cases} r(u,\pi') & u \neq v \text{ or } \pi \neq \pi' \\ max\{r(u,\pi), prog(r,u,\pi)\} & u = v \text{ and } \pi = \pi' \end{cases}
$$

Again, the operator *lift* is monotone according to $\sqsubseteq$. Every good Streett ranking r is a pre-fixpoint with respect to $lift(r, v, \pi)$ for all states $v \in V$ and permutations $\pi \in \Pi$ and every pre-fixpoint with respect to $lift(r, v, \pi)$ for all $v \in V$ and $\pi \in \Pi$ is a good Streett ranking.

By the Knaster-Tarski theorem the least pre-fixpoint (either for Streett or Rabin) exists and it can be computed by the algorithm in Fig. 3. Let $r0$ denote the following ranking. In the case of Rabin $r0$ is the ranking such that for every $v \in V$ we have $\pi(r(v)) = 12 \cdots k$ and $m(r(v)) = 0 \cdots 0$. In the case of Streett $r0$ is the ranking such that for every $v \in V$ and $\pi \in \Pi$ we have $r(v, \pi) = 0 \cdots 0$. We use the notations $lift(r, v, \pi)$ for both Rabin and Streett. In the case of Rabin we mean $lift(r, v)$.

```
RankingLifting
  Let r := r0;
  While (∃v,π s.t.  r ⊏ lift(r,v,π))
    Let r := lift(r,v,π);
  End -- While(...)
End -- RankingLifting
```

Figure 3. The lifting algorithm.

The procedure in Fig. 3 misses most of the implementation details. A naïve approach to choosing the next $v \in V$ and $\pi \in \Pi$ for performing lifting can take $O(nk!)$ for one lift. Etessami et al. supplied the necessary details for the case of parity games with 3 winning conditions [11]. In Fig. 4 we generalize their implementation to the case of Rabin and Streett ranks. As before, we handle both Rabin and Streett together. In order to handle Rabin one has to ignore the permutation π component when appropriate. Here

$C(v, \pi)$ denotes the number of successors w of v such that $r(w, \pi) = best(v, \pi)$ and $B(v, \pi)$ denotes $best(v, \pi)$.

```
foreach v ∈ V and π ∈ Π do
   B(v, π) := 0; C(v, π) := |{w : (v, w) ∈ δ}| ;
   r(v, π) := 0;
L := {v ∈ V | q ∉ L(v) and p ∈ L(v)};
while L ≠ ∅ do
   let v ∈ L;L := L \ {v};
   t := r(v);
   B(v) := best(v); C(v) := cnt(v);
   r(v) := incr_v(best(v));
   P := {w ∈ V | (w, v) ∈ ρ};
   foreach w ∈ P such that w ∉ L do
      if (w ∈ V_0 and t = B(w) and C(w) > 1)
         C(w) := C(w)−1;
      if (w ∈ V_0 and t = B(w) and C(w) = 1)
         L := L ∪ {w};
      if (w ∈ V_1 and t = B(w))
         C(w) := C(w)+1;
      if (w ∈ V_1 and t > B(w))
         L := L ∪ {w};
   endforeach
endwhile
```

Figure 4. Efficient computation of ranks.

Theorem 7 *Rabin and Streett games can be solved in time $O(mn^{k+1}kk!)$ and space $O(nk)$ for Rabin and time $O(mn^{k}kk!)$ and space $O(nkk!)$ for Streett where n is the number of states, m is the number of edges, and k is the number of pairs.*

Intuitively, the space required to hold the ranking for each state is proportional to k, which leads to the space bound of $O(nk)$ for Rabin and $O(nkk!)$ for Streett. A lift with respect to v is performed in time proportional to the number of successors of v and each comparison checks the $O(k)$ entries of the rank of a successor. Every state can be lifted at most the number of values in the respective domain. The sum of above figures leads to the stated bound. Formally, we have the following.

Proof: We start with Rabin. The space required is $O(nk)$ as we have to store the ranking for each state $v \in V$ and an entry $d \in D_R$ requires $O(k)$ space. The lifting operator can work in time $O(k \cdot out{-}deg(v))$, where $out{-}deg(v)$ is the out-degree of v. Every state can be lifted at most $|D_R|$ times. The total run time is bounded by

$$O\left(\sum_{v \in V} k \cdot out{-}deg(v) \cdot |D_R|\right) = O(km|D_R|)$$

As $|D_R| = n^{k+1}k!$ the bound follows.

For the case of Streett, the space required is $O(nkk!)$ as we have to store a value $d \in D_S$ for each state $v \in V$ and every permutation $\pi \in \Pi$. An entry $d \in D_S$ requires $O(k)$ space. The lifting operator can work in time $O(k \cdot out{-}deg(v))$. Every state and permutation can be lifted at most $|D_S|$ times. The total run time is bounded by

$$O\left(\sum_{v \in V} \sum_{\pi \in \Pi} k \cdot out{-}deg(v) \cdot |D_S|\right) = O(kmk!|D_S|)$$

As $|D_S| = n^k$ the bound follows. $\square$

As in Jurdziński's original algorithm this algorithm cannot be applied symbolically (see Section 7).

5 Recursive Algorithm

In this section we present recursive fixpoint algorithms for computing the winning sets in Rabin and Streett games. These algorithms form part of the proof of completeness of our ranking systems. There are other algorithms based on similar ideas that solve Rabin and Street games with the same complexity [17, 13]. However, we find our algorithms significantly different in one major aspect: Our algorithms are in fact a recipe for a very clean symbolic computation of the winning regions. This advantage of our algorithms led us to two results. First, our algorithms provide proofs for the completeness of the ranking system presented above. Second, the cleanliness of our algorithms enables us to use optimization techniques that were developed for symbolic fixpoint computations. The applicability of these symbolic fixpoint computation optimizations was overlooked/impossible in other solutions to Rabin and Streett games.

We comment that, as Rabin and Streett conditions are duals, the algorithms are dual. This suggests that in order to prove their correctness we could prove that both algorithms are sound and that they are dual. In order to prove that the two algorithms are dual, one would have to flatten the recursive function calls. We find it simpler to prove soundness and completeness separately.

5.1 Rabin Games

We give a recursive algorithm that solves Rabin games. Let $G=\langle V, E, \alpha\rangle$ where $\alpha=\{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ is a Rabin winning condition. An infinite play p is winning according to α if there exists some i such that $inf(p) \cap G_i \neq \emptyset$ and $inf(p) \cap R_i = \emptyset$. Intuitively, the algorithm chooses a first pair $\langle G, R\rangle$ from α, it collects recursively all the states that win according to the rest of the pairs while avoiding R. We now add states that can visit G infinitely often or get to the previously computed states. We repeat the process for the choice of other pairs as first pair. Here `cpred` denotes the control predecessor $\circledcirc$. The loop `GreatestFix(Z)` starts by setting Z to the set of all states and terminates once two consecutive rounds compute the same set of states. The loop `LeastFix(Z)` starts by setting Z to the empty set of states and terminates once two consecutive rounds compute the same set of states. Given a pair $\langle g, r\rangle$ we denote by g the set of states in g and by $\overline{r}$ the set of states in $V - R$. We freely confuse between set notation and Boolean algebra notation. Thus, given sets a and b the set $a\&b$ is the intersection of a and b and $a|b$ is the disjunction of a and b. Similarly, `true` and `false` denote the sets V and $\emptyset$ respectively.

```
Func main_Rabin(Set);
  LeastFix(Z)
   My p1 := cpred(Z);
   Z := Rabin(Set,true,p1);
  End -- LeastFix(Z)
  Return Z;
End -- Func main_Rabin(Set)

Func Rabin(Set,seqnr,right);
  My U := 0;
  Foreach (<g,r> in Set)
   My nSet := Set-<g,r>;
   GreatestFix(Y)
    My p2 := right |
   seqnr & r̄ & g & cpred(Y);
    LeastFix(X)
     My p3 := p2 |
      seqnr & r̄ & cpred(X);
      If (|nSet|=0)
       X := p3;
      Else
       X := Rabin(nSet,
              seqnr & r̄,p3);
      End -- If (|nset|=0)
     End -- LeastFix(X)
     Let Y := X;
   End -- GreatestFix(Y)
   Let U := U | Y;
  End -- Foreach (<g,r>
  Return U;
End -- Func Rabin
```

Figure 5. Recursive Algorithm for Rabin.

Theorem 8 *The algorithm in Fig. 5 computes the winning set of player 0 according to the Rabin winning condition.*

Proof: We characterize the set of states returned by the function `Rabin(S,`φ`,W)`. We show that this is the winning set in a game with a 'simpler' winning condition. We then show how the function `main_Rabin` wraps things up.

Given a set of pairs $S = \{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ we denote by $ltl\_rabin(S)$ the formula $\bigvee_{\langle G,R\rangle\in S}(\Diamond\Box\overline{R} \wedge \Box\Diamond G)$.

Claim 9 *The function* `Rabin(S,`φ`,W)` *computes the set of states winning for player 0 in the game whose winning condition is*

$$win(S,\varphi,W) = \bigvee_{\langle G,R\rangle\in S}\left[\begin{array}{ll}(\varphi\wedge\overline{R})\mathcal{U}W & \vee\\ \Box(\varphi\wedge\overline{R}\wedge\Diamond G) & \vee\\ ltl\_rabin(S-\langle G,R\rangle)\wedge\Box(\varphi\wedge\overline{R}) & \end{array}\right]$$

Proof: We prove the claim by induction on the number of pairs in S. Suppose $S = \{\langle G, R\rangle\}$, then $Rabin(S,\varphi,W)$ returns the following fixpoint.

$$\nu Y \mu X(W \vee \varphi\wedge\overline{R}\wedge G\wedge\circledcirc Y \vee \varphi\wedge\overline{R}\wedge\circledcirc X)$$

Let $\hat{Y}$ denote the set computed by this fixpoint. Let $X_0 = \emptyset$ and let

$$X_{i+1} = W \vee (\varphi\wedge\overline{R}\wedge G\wedge\circledcirc\hat{Y}) \vee (\varphi\wedge\overline{R}\wedge\circledcirc X_i)$$

It follows that $\hat{Y} = \bigcup_{i=1}^{\infty} X_i$. We associate every $v \in \hat{Y}$ a rank that is the minimal i such that $v \in X_i$.

If $v \in X_1$ then either $v \in W$ or $V \models \varphi\wedge\overline{R}\wedge G$ and player 0 can force the play in the next move to some state in $\hat{Y}$. If $v \in X_i$ for $i > 1$ then $V \models \varphi\wedge\overline{R}$ and player 0 can force the play to some state in X_{i-1}. It follows that player 0 has a strategy to win $[(\varphi\wedge\overline{R})\mathcal{U}W] \vee \Box(\overline{R}\wedge\varphi\wedge\Diamond G)$. So in the case that $|S| = 1$ the claim is sound.

We show that in the case that $|S| = 1$ the claim is complete. Let W_0 denote the winning set for player 0 according to the winning condition $win(S,\varphi,W)$. Let $\hat{Y}$ denote some set such that $W_0 \subseteq \hat{Y}$. We show that every state from which player 0 wins a game with a simpler winning condition is maintained in the computation of the greatest fixpoint. As the winning condition $win(S,\varphi,W)$ implies this simpler winning condition it follows that the greatest fixpoint does not loose winning states according to win. Consider the following winning condition.

$$\psi = [(\varphi\wedge\overline{R})\mathcal{U}W] \vee [(\varphi\wedge\overline{R})\mathcal{U}(\varphi\wedge\overline{R}\wedge G\wedge\circledcirc\hat{Y})]$$

We show that every state winning according to ψ is maintained in the next iteration of the greatest fixpoint. Let X_i for $i \geq 0$ be the sets defined above. For a state v from which player 0 wins according to ψ let i denote the maximal number of steps that are taken until a state in W or in $(\varphi\wedge p\wedge q\wedge\circledcirc\hat{Y})$ is reached. If $i = 0$ then clearly $v \in X_0$. If $i > 0$ then if v is a state of player 0 there exists a successor of v whose distance from $W \vee (\varphi\wedge p\wedge q\wedge\circledcirc\hat{Y})$ is at most $i-1$. This successor is in X_{i-1} by induction and $v \in X_i$. If v is a state of player 1 then all successors of v are in X_{i-1}. This completes the proof of the base case of the induction.

Suppose that the claim is true for sets S of size i. We prove the claim for sets of size $i+1$. We concentrate on one pair $\langle G, R\rangle \in S$ and denote $S' = S - \langle G, R\rangle$. The largest fixpoint in Rabin(S, φ, W) computes the following set.

$$\nu Y \mu X \left[Rabin \left(S', \varphi \wedge \overline{R}, \begin{pmatrix} W & \vee \\ (\varphi \wedge \overline{R} \wedge G \wedge \bigcirc Y) \vee & \\ (\varphi \wedge \overline{R} \wedge \bigcirc X) & \end{pmatrix} \right) \right]$$

As before let $\hat{Y}$ denote the result of this fixpoint, let X_0 be the empty set and let

$$X_{i+1} = Rabin \left(S', \varphi \wedge \overline{R}, \begin{pmatrix} W & \vee \\ (\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y}) & \vee \\ (\varphi \wedge \overline{R} \wedge \bigcirc X_i) & \end{pmatrix} \right)$$

from the induction assumption

$$x_{i+1} = win \left(S', \varphi \wedge \overline{R}, \begin{pmatrix} W & \vee \\ (\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y}) & \vee \\ (\varphi \wedge \overline{R} \wedge \bigcirc X_i) & \end{pmatrix} \right)$$

Suppose that $v \in X_1$. By induction, from v player 0 wins with the winning condition

$$\bigvee_{\langle G',R'\rangle \in S'} \begin{bmatrix} ((\varphi \wedge \overline{R} \wedge \overline{R}')\mathcal{U}(W \vee (\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y}))) \vee \\ \Box(\varphi \wedge \overline{R} \wedge \overline{R}' \wedge \Diamond G') \qquad \vee \\ (ltl\_rabin(S' - \langle G', R'\rangle) \wedge \Box(\varphi \wedge \overline{R} \wedge \overline{R}')) \end{bmatrix}$$

which is equivalent to

$$\bigvee_{\langle G',R'\rangle \in S'} \begin{bmatrix} ((\varphi \wedge \overline{R} \wedge \overline{R}')\mathcal{U}(\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y})) & \vee \\ ((\varphi \wedge \overline{R} \wedge \overline{R}')\mathcal{U}W) & \vee \\ \Box(\varphi \wedge \overline{R} \wedge \overline{R}' \wedge \Diamond G') & \vee \\ (ltl\_rabin(S' - \langle G', R'\rangle) \wedge \Box(\varphi \wedge \overline{R} \wedge \overline{R}')) & \end{bmatrix}$$

So player 0 can either (a) force the game to a state in W while maintaining $\varphi \wedge \overline{R}$, (b) force the game to a state that satisfies $\varphi \wedge \overline{R} \wedge G$ while maintaining $\varphi \wedge \overline{R}$ and then force the game in the next move to $\hat{Y}$ or (c) win according to the rest of the condition.

Suppose that $v \in X_i$ for $i > 1$. By induction, from v player 0 wins with the winning condition

$$\bigvee_{\langle G',R'\rangle \in S'} \begin{bmatrix} \left(\begin{pmatrix} \varphi \wedge \\ \overline{R} \wedge \\ \overline{R}' \end{pmatrix} \mathcal{U} \begin{pmatrix} W & \vee \\ (\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y}) & \vee \\ (\varphi \wedge \overline{R} \wedge \bigcirc X_{i-1}) & \end{pmatrix} \right) & \vee \\ \Box(\varphi \wedge \overline{R} \wedge \overline{R}' \wedge \Diamond G') & \vee \\ (ltl\_rabin(S' - \langle G', R'\rangle) \wedge \Box(\varphi \wedge \overline{R} \wedge \overline{R}')) & \end{bmatrix}$$

which is equivalent to

$$\bigvee_{\langle G',R'\rangle \in S'} \begin{bmatrix} ((\varphi \wedge \overline{R} \wedge \overline{R}')\mathcal{U}(\varphi \wedge \overline{R} \wedge \bigcirc X_{i-1})) & \vee \\ ((\varphi \wedge \overline{R} \wedge \overline{R}')\mathcal{U}(\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y})) & \vee \\ ((\varphi \wedge \overline{R} \wedge \overline{R}')\mathcal{U}W) & \vee \\ \Box(\varphi \wedge \overline{R} \wedge \overline{R}' \wedge \Diamond G') & \vee \\ (ltl\_rabin(S' - \langle G', R'\rangle) \wedge \Box(\varphi \wedge \overline{R} \wedge \overline{R}')) & \end{bmatrix}$$

So player 0 has a strategy that either (a) forces the game to a state in W while maintaining $\varphi \wedge \overline{R}$, (b) forces the game to a state in $\varphi \wedge \overline{R} \wedge G \wedge \bigcirc \hat{Y}$ while maintaining $\varphi \wedge \overline{R}$, (c) forces the game to X_{i-1} while maintaining $\varphi \wedge \overline{R}$, or (d) wins according to

$$\psi := \bigvee_{\langle G',R'\rangle \in S'} \begin{bmatrix} \Box(\varphi \wedge \overline{R}\, wedge \overline{R}' \wedge \Diamond G') & \vee \\ \begin{pmatrix} ltl\_rabin(S - \langle G', R'\rangle) & \wedge \\ \Box(\varphi \wedge \overline{R} \wedge \overline{R}') & \end{pmatrix} & \end{bmatrix}$$

We combine these strategies as follows. Consider a state $v \in \hat{Y}$. Let i be the minimal such that $v \in X_i$. Player 0 applies the ith strategy. Either the play remains in X_i indefinitely or it reaches X_{i-1} and player 0 switches to the $i-1$th strategy. If while playing according to some strategy the play reaches $\varphi \wedge \overline{R} \wedge G$ then player 0 chooses some successor in $\hat{Y}$ and continues with the appropriate strategy. Consider an infinite play according to the combination of the strategies as explained above. Either for some i the play stays indefinitely in X_i and wins according to ψ or infinitely often the play reaches X_1 and wins according to $\Box(\varphi \wedge \overline{R} \wedge \Diamond G)$.

Every play is won according to one of the following.

- The play starts with a finite prefix of $\varphi \wedge \overline{R}$ and stays eventually always within some X_i and wins according to ψ.
- The play visits X_1 infinitely often and satisfies $\Box(\varphi \wedge \overline{R} \wedge \Diamond G)$.
- The play gets to W along a $\varphi \wedge \overline{R}$ path.

This means that the Rabin player wins according to the following condition.

$$\bigvee_{\langle G',R'\rangle \in S'} \begin{bmatrix} \Box(\overline{R} \wedge \varphi \wedge \Diamond G) & \vee \\ (\varphi \wedge \overline{R})\mathcal{U}W & \vee \\ (\Diamond \Box \overline{R}' \wedge \Box \Diamond G') \wedge \Box(\overline{R} \wedge \varphi) & \vee \\ (ltl\_rabin(S' - \langle G', R'\rangle) \wedge \Box(\varphi \wedge \overline{R})) & \end{bmatrix}$$

or equivalently

$$\begin{matrix} \Box(\overline{R} \wedge \varphi \wedge \Diamond G) & \vee \\ (\varphi \wedge \overline{R})\mathcal{U}W & \vee \\ (ltl\_rabin(S') \wedge \Box(\varphi \wedge \overline{R})) & \end{matrix}$$

We note that the greatest fixpoint in Rabin is nested in a loop going over all pairs in S. We conclude that the winning condition is of the wanted form.

We now prove the completeness of the induction step. We show that every iteration of the greatest fixpoint maintains all the states winning according to a simpler winning condition ψ. As $win(S, \varphi, W)$ implies ψ it follows that every state winning according to win remains in the greatest fixpoint. Consider some pair $\langle G, R\rangle \in S$ and denote $S' = S - \langle G, R\rangle$. Let W_0 denote the winning set for player 0 according to the disjunct of $\langle G, R\rangle$ in $win(S, \varphi, W)$. Let $\hat{Y}$ denote some set such that $W_0 \subseteq \hat{Y}$. We show that every state from which player 0 wins the game whose winning condition is

$$\psi = \begin{array}{ll} (\varphi \wedge \overline{R})\mathcal{U}(\varphi \wedge \overline{R} \wedge G \wedge \diamondsuit \hat{Y}) & \vee \\ (\varphi \wedge \overline{R})\mathcal{U}W & \vee \\ (ltl\_rabin(S') \wedge \Box(\varphi \wedge \overline{R})) & \end{array}$$

is maintained by the greatest fixpoint.

Denote the winning region for player 0 according to the winning condition ψ by T. We analyze the form of T, our methods remind the methods in [18]. We show that as long as $\hat{X}$ is not equal to T the equation $rabin(S', \varphi \wedge \overline{R}, W \vee (\varphi \wedge \overline{R} \wedge G \wedge \diamondsuit \hat{Y}) \vee (\varphi \wedge \overline{R} \wedge \diamondsuit \hat{X}))$ increases the size of X. As T is finite it follows that eventually the minimal fixpoint equals T.

It is clear that every state in T satisfies $\varphi \wedge \overline{R}$. Suppose that there exists some state $v \in T - \hat{X}$ such that player 0 can control the play to reach $\hat{X}$ in one step then v is included in the next value of the fixpoint. Suppose that no such state exists. We show that there exists a state from which player 0 wins according to $win(S', \varphi \wedge \overline{R}, W \vee (\varphi \wedge \overline{R} \wedge \hat{X}))$. That is, player 0's strategy on $T - \hat{X}$ maintains $\Box(\varphi \wedge \overline{R})$ and wins according to $ltl\_rabin(S')$. We show that there exists a node $v \in T - \hat{X}$ such that player 0's winning strategy maintains $\Box \overline{R}'$ for some pair $\langle G', R'\rangle \in S'$. This state is included in $win(S', \varphi \wedge \overline{R}, W \vee (\varphi \wedge \overline{R} \wedge \diamondsuit \hat{X}))$ by the induction assumption. Suppose that there does not exist a state $v \in T - \hat{X}$ such that for some $\langle G', R'\rangle \in S'$ player 0's winning strategy maintains $\Box \overline{R}'$ on all plays continuing from v. Let $\langle G', R'\rangle$ be the first pair in S'. By assumption there does not exist a state from which player 0 maintains $\Box \overline{R}'$. Let $v_0 \in T - \hat{X}$ be some state such that $v_0 \models R'$. We recall that player 0 cannot force an immediate visit to $\hat{X}$. There exists a successor v_1 that is either chosen by player 0 (in case that v_0 is a state of player 0) or it is some successor of v_0 in $T - \hat{X}$ (in case that v_0 is a state of player 1). We construct by induction an infinite path in $T - \hat{X}$ that visits R' for every $\langle G', R'\rangle \in S'$ infinitely often. This path cannot be winning according to ψ. We conclude that there exists a node v from which player 0's winning strategy maintains $\Box \overline{R}'$ for some $\langle G', R'\rangle \in S'$. This state is winning according to $win(S', \varphi \wedge \overline{R}, W \vee (\varphi \wedge \overline{R} \wedge \diamondsuit \hat{X}))$ and it is included in the next iteration of the fixpoint. This concludes completeness of the induction step. □

We handle the function `main_Rabin`. From the previous proof it immediately follows that every state returned by `main_Rabin` is winning for player 0. We have to show that every state winning for player 0 is included. Similar to the completeness proof above, we analyze the winning region for player 0 in the Rabin game. We claim that there exists a region in the winning region of player 0 that satisfies $\Box \overline{R}$ for some $\langle G, R\rangle \in S$. Such a region satisfies the condition $win(S, true, \emptyset)$. It follows that it is returned in a call to `Rabin(S,true,∅)`. Then the minimal fixpoint collects all states that can reach these regions in a finite number of states and collects other such regions. As before, the game is finite so it is eventually depleted.

Formally, assume that W is the set of states computed by the minimal fixpoint in `main_Rabin`. Assume further that W_0 is the set of winning states for player 0 in the Rabin game and that $W_0 - W \neq \emptyset$. Suppose that there exists some state v in $W_0 - W$ such that player 0 can control the play to reach W in one step. Then v is included in the next value of the fixpoint. Suppose that no such state exists. Then we show that there exists a state from which player 0 wins according to $win(S, true, W)$. That is, player 0's strategy on $W_0 - W$ maintains $\Box \overline{R}$ for some pair $\langle G, R\rangle \in S$ and in addition wins according to the Rabin condition. Suppose that there does not exist a state $v \in W_0 - W$ such that for some $\langle G, R\rangle \in S$ player 0's winning strategy maintains $\Box \overline{R}$. Let v_0 be some state such that $v_0 \models R_1$. We recall that player 0 cannot force an immediate visit to W. There exists a successor v_1 that is either chosen by player 0 or it is some successor of v_0 in $W_0 - W$. We construct by induction an infinite path in $W_0 - W$ that visits R for every pair $\langle G, R\rangle \in S$ infinitely often. This path cannot be winning according to the Rabin condition and we conclude that a state from which player 0's winning strategy maintains $\Box \overline{R}$ exists. This state is included in the next iteration of the fixpoint and eventually the minimal fixpoint equals W_0. □

5.2 Streett Games

We give a recursive algorithm that solves Streett games. Let $G = \langle V, E, \alpha\rangle$ where $\alpha = \{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ is a Streett winning condition. An infinite play p is winning according to α if forall i we have $inf(p) \cap G_i \neq \emptyset$ implies $inf(p) \cap R_i \neq \emptyset$. Intuitively, the algorithm chooses a pair $\langle G, R\rangle$ in α, it collects all states that eventually avoid G states while making sure recursively that all other pairs are satisfied. We then add states that can visit R infinitely often and do the same for all other pairs.

Theorem 10 *The algorithm in Fig. 6 computes the winning set of player 0 according to the Streett winning condition.*

```
Func main_Streett(Set)
 If (|nSet|=0)
    Return m_Streett(true,false);
 Return Streett(Set,true,false);
End -- Func main_Streett(Set)

Func Streett(Set,seqng,right)
 GreatestFix(Z)
  Foreach (<g,r> in Set)
   My nSet := Set-<g,r>;
   My p1 := right |
     seqp & r & cpred(Z);
   LeastFix(Y)
    My p2 := p1 |
        seqng & cpred(Y);
    If (|nSet|=0)
     Y := m_Streett(
            seqng & g̅,p2);
    Else
     Y := Streett(nSet,
            seqng & g̅,p2);
    End -- If (|nSet|=0)
   End -- LeastFix(Y)
   Z := Y;
  End -- Foreach (<g,r>
 End -- GreatestFix(Z)
 Return Z;
End -- Streett

Func m_Streett(seqng,right)
GreatestFix(X)
  X := right |
        seqng & cpred(X);
End -- GreatestFix(X)
Return X;
End -- m_Streett
```

Figure 6. Recursive Algorithm for Streett.

Proof: We characterize the set of states returned by the function `main_Streett(S)`. We show that this is the winning set in a game with a 'simpler' winning condition.

Given a set of pairs $S = \{\langle G_1, R_1\rangle, \ldots, \langle G_k, R_k\rangle\}$ we denote the formula $\bigwedge_{\langle G,R\rangle \in S}(\Box\Diamond G \rightarrow \Box\Diamond R)$ by $ltl\_streett(S)$.

Claim 11 *The function* `m_Streett`*(φ,W) computes the set of states winning for player 0 in the game whose winning condition is $\varphi \mathcal{W} W \vee \Box\,\varphi$.*

Proof: The function $\texttt{m\_Streett}(\varphi, W)$ computes the fixpoint $\nu X(W \vee \varphi \wedge \odot X)$. This is exactly the set of states that satisfy $\varphi \mathcal{U} W \vee \Box\,\varphi$. $\Box$

Claim 12 *The function* `Streett(S,`φ`,W)` *computes the set of states winning for player 0 in the game whose winning condition is*

$$win(S,\varphi,W) = (\varphi \mathcal{U} W) \quad \vee \quad \bigwedge_{\langle G,R\rangle\in S} \left[\begin{array}{l} \Box(\varphi \wedge \Diamond R) \qquad \vee \\ \varphi \mathcal{U} \left(\begin{array}{l} \Box(\varphi \wedge \overline{G}) \qquad \wedge \\ ltl\_streett(S - \langle G,R\rangle) \end{array} \right) \end{array} \right]$$

Proof: We prove the claim by induction on the number of pairs in S. Suppose $S = \{\langle G, R\rangle\}$, then $Streett(S, \varphi, W)$ computes the following fixpoint.

$$\nu Z \mu Y (\texttt{m\_Streett}(\varphi \wedge \overline{G}, \left(\begin{array}{l} W \qquad \vee \\ \varphi \wedge R \wedge \odot Z \vee \\ \varphi \wedge \odot Y \end{array} \right)))$$

Let $\hat{Z}$ denote the set computed by the greatest fixpoint. Let $Y_0 = \emptyset$ and let

$$Y_{i+1} = \texttt{m\_Streett}(\varphi \wedge \overline{G}, \left(\begin{array}{l} W \qquad \vee \\ \varphi \wedge R \wedge \odot \hat{Z} \vee \\ \varphi \wedge \odot Y_i \end{array} \right))$$

For every state $v \in \hat{Z}$ let $r(v)$ be the minimal i such that $v \in Y_i$.

Consider a state v such that $r(v) = 1$. By induction player 0 wins from v according to

$$\Box(\varphi \wedge \overline{G}) \vee (\varphi \wedge \overline{G}) \mathcal{U} \left(\begin{array}{l} W \qquad \vee \\ \varphi \wedge R \wedge \odot \hat{Z} \end{array} \right)$$

So there exists a strategy such that player either (a) reaches W while staying in φ states, (b) reaches $\varphi \wedge R \wedge \odot \hat{Z}$ while staying in φ states, or (c) the play is infinite and it is always in $\varphi \wedge \overline{G}$. Consider a state v such that $r(v) = i > 1$. Player 0 wins from v according to

$$\Box(\varphi \wedge \overline{G}) \vee (\varphi \wedge \overline{G}) \mathcal{U} \left(\begin{array}{l} W \qquad \vee \\ \varphi \wedge R \wedge \odot \hat{Z} \vee \\ \varphi \wedge \odot Y_{i-1} \end{array} \right)$$

So there exists a strategy such that player 0 either (a) reaches W while staying in $\varphi \wedge \overline{G}$ states, (b) reaches a state with lower rank or reaches $\varphi \wedge R$ while staying in φ states, or (c) the play is infinite and it is always in $\varphi \wedge \overline{G}$ states.

We now combine these strategies to prove the soundness of the claim in case that $|S| = 1$. In states whose rank is i player 0 player the ith strategy. While playing according to some strategy and getting to a state in $\varphi \wedge R \wedge \odot \hat{Z}$, player 0 chooses some successor in $\hat{Z}$ and the rank may increase arbitrarily. Every play either stays within some Y_i form some stage onwards and continues indefinitely according to the ith strategy or infinitely often switches between the strategies. In the first case, the play fulfills $\varphi\mathcal{U}\,\square(\varphi\wedge\overline{G})$, which implies $\varphi\mathcal{U}(ltl\_streett(S)\wedge\square(\varphi))$. In the second case, the play fulfills $\square(\varphi\wedge\Diamond R)$. Soundness of the case that $|S|=1$ follows.

We prove completeness in the case that $|S| = 1$. Let W_0 denote the winning set of player 0 according to $win(S,\varphi,W)$. Let $\hat{Z}$ denote some set such that $W_0\subseteq\hat{Z}$. We show that every state from which player 0 wins the game according to

$$\psi = \begin{array}{ll}\varphi\mathcal{U}(W\vee(\varphi\wedge R\wedge\circledcirc\hat{Z})) & \vee\\ \square(\varphi\wedge\overline{G}) & \end{array}$$

is maintained by the fixpoint. Clearly, a state winning according to $win(S,\varphi,W)$ is winning according to ψ.

Denote the winning region for player 0 according to ψ by T. We analyze the form of T. We show that as long as $\hat{Y}$ is not equal to T the function call m_Streett($\varphi\wedge\overline{G},W\vee(\varphi\wedge R\wedge\hat{Z}\vee\varphi\wedge\circledcirc\hat{Y})$) increases the size of Y. As T is finite it follows that eventually the minimal fixpoint equals T.

It is clear that every state in T satisfies φ. Suppose that there exists some state $v\in T$ such that player 0 can control the play to reach $\hat{Y}$ in one step, then v is included in the next value of the fixpoint. Suppose that no such state exists. We show that there exists a state from which player 0 wins according to $\square(\varphi\wedge\overline{G})$.

Suppose that such a state does not exist. That is, there does not exist a state for which player 0's winning strategy maintains $\varphi\wedge\overline{G}$. Let v_0 be some state such that $v\models G$. We recall that player 0 cannot force an immediate visit to $\hat{Y}$. There exists a successor v_{2i+1} of v_{2i} that is either chosen by player 0 (in case that v_{2i} is a state of player 0) or is some successor of v_{2i} in $T-\hat{Y}$ (in case that v_{2i} is a state of player 1). By assumption player 0 does not maintain $\square\,\overline{G}$ and there exists a node v_{2i+2} that is reachable from v_{2i+1} using player 0's winning strategy such that $v_{2i+2}\models G$. By induction we construct an infinite path in $T-\hat{Y}$ that respects player 0's winning strategy and visits G infinitely often. This path cannot be winning according to ψ. We conclude that there exists a node v from which player 0's winning strategy maintains $\square\,\overline{G}$. This state is in addition winning according to ψ. We conclude that this state is winning according to $\square(\varphi\wedge\overline{G})$ and that it is included in the next iteration of the fixpoint. This concludes completeness of the claim in case that $|S|=1$.

The induction step is similar to the proof of the induction base. Suppose that the claim is true for sets S of size i. We prove the claim for sets of size $i+1$. Let $\hat{Z}$ denote the set computed by the greatest fixpoint. It follows that for every $\langle G,R\rangle\in S$ we have

$$\hat{Z} = \mu Y\left[Streett\left(S-\langle G,R\rangle,\varphi\wedge\overline{G},\begin{pmatrix}W & \vee\\ (\varphi\wedge R\wedge\circledcirc\hat{Z})\vee & \\ (\varphi\wedge\circledcirc Y) & \end{pmatrix}\right)\right]$$

We concentrate on some $\langle G,R\rangle\in S$ and denote $S'=S-\langle G,R\rangle$. Let $Y_0=\emptyset$ and let

$$Y_{i+1}=Streett(S',\varphi\wedge\overline{G},W\vee(\varphi\wedge R\wedge\circledcirc\hat{Z})\vee(\varphi\wedge\circledcirc Y_i))$$

For every state $v\in\hat{Z}$ let $r(v)$ be the minimal i such that $v\in Y_i$.

Consider a state v such that $r(v)=1$. By induction player 0 wins from v according to

$$\begin{array}{l}\varphi\mathcal{U}(W\vee(\varphi\wedge R\wedge\circledcirc\hat{Z})) \hfill\vee\\ \bigwedge_{\langle G',R'\rangle\in S'}\left[\begin{array}{l}\square(\varphi\wedge\overline{G}\wedge\Diamond R') \hfill\vee\\ (\varphi\wedge\overline{G})\mathcal{U}\begin{pmatrix}\square(\varphi\wedge\overline{G}) & \wedge\\ ltl\_streett(S'-\langle G',R'\rangle) & \end{pmatrix}\end{array}\right]\end{array}$$

or equivalently

$$\begin{array}{l}\varphi\mathcal{U}(\varphi\wedge R\wedge\circledcirc\hat{Z}) \hfill\vee\\ \varphi\mathcal{U}W \hfill\vee\\ \bigwedge_{\langle G',R'\rangle\in S'}\left[\begin{array}{l}\square(\varphi\wedge\overline{G}\wedge\Diamond R') \hfill\vee\\ (\varphi\wedge\overline{G})\mathcal{U}\begin{pmatrix}\square(\varphi\wedge\overline{G}) & \wedge\\ ltl\_streett(S'-\langle G',R'\rangle) & \end{pmatrix}\end{array}\right]\end{array}$$

So there exists a strategy such that player 0 either (a) reaches W while staying in φ states, (b) reaches $\varphi\wedge R\wedge\circledcirc\hat{Z}$ while staying in φ states, or (c) the play is infinite and it is always in $\varphi\wedge\overline{G}$ states while satisfying the rest of the Streett pairs.

Consider a state v such that $r(v)=i>1$. By induction player 0 wins from v according to

$$\begin{array}{l}\varphi\mathcal{U}(\varphi\wedge\circledcirc Y_{i-1}) \hfill\vee\\ \varphi\mathcal{U}(\varphi\wedge R\wedge\circledcirc\hat{Z}) \hfill\vee\\ \varphi\mathcal{U}W \hfill\vee\\ \bigwedge_{\langle G',R'\rangle\in S'}\left[\begin{array}{l}\square(\varphi\wedge\overline{G}\wedge\Diamond R') \hfill\vee\\ (\varphi\wedge\overline{G})\mathcal{U}\begin{pmatrix}\square(\varphi\wedge\overline{G}) & \wedge\\ ltl\_streett(S'-\langle G',R'\rangle) & \end{pmatrix}\end{array}\right]\end{array}$$

So there exists a strategy such that player 0 either (a) reaches W while staying in φ states, (b) reaches a state with lower rank or reaches $\varphi\wedge R$ while staying in φ states, or (c) the play is infinite and it is always in $\varphi\wedge\overline{G}$ states while satisfying the rest of the Streett pairs.

We now combine these strategies to prove the soundness of the induction step. In states whose rank is i player 0 plays the ith strategy. When playing according to some strategy and getting to a state in $\varphi\wedge R\wedge\circledcirc\hat{Z}$, player 0 chooses some successor and the rank may increase arbitrarily. Every play either stays within some Y_i from some stage onwards and continues indefinitely according to the ith strategy or infinitely often switches between the strategies. In the first case, the play fulfills

$$\varphi\mathcal{U} \bigwedge_{\langle G',R'\rangle\in S'} \begin{bmatrix} \Box(\varphi\wedge\overline{G}\wedge\Diamond R') & \vee \\ (\varphi\wedge\overline{G})\mathcal{U}\left(\begin{array}{ll} \Box(\varphi\wedge\overline{G}) & \wedge \\ ltl\_streett(S'-\langle G',R'\rangle) & \end{array}\right) & \end{bmatrix}$$

which implies $\varphi\mathcal{U}(ltl\_streett(S')\wedge\Box(\varphi))$. In the second case, the play fulfills $\Box(\varphi\wedge\Diamond R)$. Soundness follows.

We now prove the completeness of the induction step. Let W_0 denote the winning set of player 0 according to $win(S,\varphi,W)$. Let $\hat{Z}$ denote some set such that $W_0\subseteq\hat{Z}$. We concentrate on some pair $\langle G,R\rangle\in S$ and denote $S'=S-\langle G,R\rangle$. We show that every state from which player 0 wins the game according to

$$\psi = \begin{array}{ll} \varphi\mathcal{U}(W\vee(\varphi\wedge R\wedge\diamondsuit\hat{Z})) & \vee \\ \varphi\mathcal{U}\left[\begin{array}{ll} \Box(\varphi\wedge\overline{G}) & \wedge \\ ltl\_streett(S-\langle G,R\rangle) & \end{array}\right] & \end{array}$$

is maintained by the fixpoint. Clearly, a state winning according to $win(S,\varphi,W)$ is winning according to ψ.

Denote the winning region for player 0 according to ψ by T. We analyze the form of T. We show that as long as $\hat{Y}$ is not equal to T the equation $streett(S',\varphi\wedge\overline{G},W\vee(\varphi\wedge\diamondsuit\hat{Y}))$ increases the size of Y. As T is finite it follows that eventually the minimal fixpoint equals T.

It is clear that every state in T satisfies φ. Suppose that there exists some state $v\in T$ such that player 0 can control the play to reach $\hat{Y}$ in one step, then v is included in the next value of the fixpoint. Suppose that no such state exists. We show that there exists a state from which player 0 wins according to $win(S',\varphi\wedge\overline{G},W\vee(\varphi\wedge\diamondsuit\hat{Y}))$.

As before it is sufficient to prove that there exists a state v such that player 0's winning strategy for ψ maintains $\Box(\varphi\wedge\overline{G})$. Combining $\Box(\varphi\wedge\overline{G})$ with ψ gives us a region winning with respect to $win(S',\varphi\wedge\overline{G},W\vee(\varphi\wedge\diamondsuit\hat{Y}))$. Suppose that there does not exists a state v such that player 0's winning strategy maintains $\Box\overline{G}$. Let v_0 be some state such that $v_0\models G$. We recall that player 0 cannot force an immediate visit to $\hat{Y}$. There exists a successor v_{2i+1} of v_{2i} that is either chosen by player 0 (in case that v_{2i} is a state of player 0) or is some successor of v_{2i} in $T-\hat{Y}$ (in case that v_{2i} is a state of player 1). By assumption player 0 does not maintain $\Box\overline{G}$ and there exists a node v_{2i+2} that is reachable from v_{2i+1} using player 0's winning strategy such that $v_{2i+2}\models G$. By induction we construct an infinite path in $T-\hat{Y}$ that respects player 0's winning strategy and visits G infinitely often. This path cannot be winning according to ψ. We conclude that there exists a node v from which player 0's winning strategy maintains $\Box\overline{G}$. This state is in addition winning according to ψ. We conclude that this state is winning according to $win(S',\varphi\wedge\overline{G},W\vee(\varphi\wedge\diamondsuit\hat{Y}))$ and that it is included in the next iteration of the fixpoint. This concludes completeness of the induction step. □

□

From Theorems 8 and 10 it is easy to derive the following bounds. A greatest or least fixpoint collects at least one state in every iteration and hence cannot be repeated more than n times. The inner most fixpoint can be computed in time proportional to m where m is the number of transitions.

Corollary 13 *Rabin and Streett games can be solved symbolically in time $O(mn^{2k}k!)$ where n is the number of states, m is the number of transitions, and k is the number of pairs of the winning condition.*

We stress that these algorithms are not important by themselves. Indeed, the same complexity is achieved by other similar algorithms [17, 13]. They are used to establish the completeness of the ranks presented in Section 3. Efficient computation of these ranks leads to algorithms with improved complexity.

6 Fast Symbolic Computation

In this section we generalize the method of Long et al. for accelerating the evaluation of fixpoints [19]. Long et al. show that by maintaining the intermediate values of the fixpoint, they can use these values to start the computation of future fixpoints not from minimal or maximal values but rather from better approximations. They show that with these approximations the worst time complexity of the fixpoint computation is reduced to the square root of the original. Unfortunately, the memory consumption amounts to the other square root.

The acceleration works very similarly for Rabin and Streett games. We explain it here for the case of Rabin. The case of Streett is identical but for the order of the indices. Consider the algorithm in Fig. 5. We add a counter to each of the fixpoints. To each of the minimal fixpoints we add a counter i. It is initialized to 0 in the first visit to the command `LeastFix` and incremented by 1 in every subsequent visit. Similarly, to each of the maximal fixpoint we add a counter p. It is initialized to 0 in the first visit to the command `GreatestFix` and incremented by 1 in every subsequent visit. Consider an active copy of the function `Rabin` with $l-1$ copies of `Rabin` on the store. Suppose that the active copy of `Rabin` is found in line 4. Let $i_0\cdots i_{l-1}$ be the values of the counters i associated with the least fixpoints in the copies of `Rabin` on the stack (where i_0 is the counter in the function `main_Rabin`). Let $p_1\cdots p_l$ be the values of the counters p associated with the greatest fixpoints in the copies of `Rabin` on the stack. Let $j_1 \cdots j_l$ denote the number of pairs handled by the different copies of `Rabin`. We set $Y(i_0, \cdots i_{l-1}, p_1 \cdots p_l, j_1 \cdots j_l)$ to be the value of Y when the counter p is set to p_l. When the active copy of `Rabin` is found in line 6 Then the sequence $i_0 \cdots i_l$ includes also the value of the counter i in the active copy of `Rabin`. We set $X(i_0 \cdots i_l, p_1 \cdots p_l, j_1 \cdots j_l)$ to be the value of X when the counter i is set to i_l.

Given sequences $\alpha = i_0 \cdots i_{l-1}$, $\beta = p_1 \cdots p_l$, and $\gamma = j_1 \cdots j_l$ and $\alpha' = i'_0 \cdots i'_{l-1}$, $\beta' = p'_1 \cdots p'_l$, and $\gamma' = j'_1 \cdots j'_l$ we say that $\alpha\beta\gamma <_\nu \alpha'\beta'\gamma'$ if $\alpha = \alpha'$, $\gamma = \gamma'$ and $\beta < \beta'$ according to the lexicographic order. Similarly, given $\alpha = i_0 \cdots i_l$, $\beta = p_1 \cdots p_l$, and $\gamma = j_1 \cdots j_l$ and $\alpha' = i'_0 \cdots i'_l$, $\beta' = p'_1 \cdots p'_l$, and $\gamma' = j'_1 \cdots j'_l$ we say that $\alpha\beta\gamma <_\mu \alpha'\beta'\gamma'$ if $\beta = \beta'$, $\gamma = \gamma'$ and $\alpha < \alpha'$ according to the lexicographic order. For a fixed $\alpha = i_0 \cdots i_{l-1}$ and $\gamma = j_1 \cdots j_l$, the ordering $<_\nu$ is a total order on l-tuples. Similarly, for a fixed $\beta = j_1 \cdots j_l$ and $\gamma = j_1 \cdots j_l$, the ordering $<_\mu$ is a total order on l-tuples.

For every $\alpha = i_0 \cdots i_{l-1}$, $\beta = p_0 \cdots p_{l-1}$, and $\gamma = j_1 \cdots j_l$ the maximal value p such that $Y(\alpha, \beta p, \gamma)$ is defined is a greatest fixpoint value. Long et al. show that $Y(\alpha, \beta p, \gamma)$ is contained in every set $Y(\alpha, \beta', \gamma)$ such that $\beta' < \beta p$. It follows that the computation of $Y(\alpha, \beta 0, \gamma)$ (which leads to the computation of $Y(\alpha, \beta p, \gamma)$) can start from the minimal set $Y(\alpha, \beta', \gamma)$ such that $\beta' < \beta 0$. Consider now the values of the inner-most greatest fixpoint. That is, the values $Y(\alpha, \beta, \gamma)$ where $|\beta| = k$. It follows that for every value of α and γ there are at most n different values for $Y(\alpha, \beta, \gamma)$.

Dually, for every $\alpha = i_0 \cdots i_{l-1}$, $\beta = p_0 \cdots p_l$, and $\gamma = j_1 \cdots j_l$ the maximal value i such that $X(\alpha i, \beta, \gamma)$ is defined is a least fixpoint value. Long et al. show that $X(\alpha i, \beta, \gamma)$ contains every set $X(\alpha', \beta, \gamma)$ such that $\alpha' < \alpha i$. It follows that the computation of $X(\alpha 0, \beta, \gamma)$ (which leads to the computation of $X(\alpha i, \beta, \gamma)$ can start from the maximal set $X(\alpha', \beta, \gamma)$ such that $\alpha' < \alpha 0$). Consider now the values of the inner-most least fixpoint. That is, the values $X(\alpha, \beta, \gamma)$ where $|\alpha| = k$. If follows that for every value of β and γ there are at most n different values for $X(\alpha, \beta, \gamma)$.

The computation of the inner-most least fixpoint dominates the computation time. It follows that the computation can be concluded in time $O(n^{k+1}k!)$. However, we have to store the Y values for every possible value of α and γ. Notice, that the β values are implicit in every point of the computation. We just have to store the best value for α and γ. Similarly, we store the X values for every possible value of β and γ. Thus, the memory required by the algorithm is $O(n^{k+1}k!)$. Formally, we have the following.

Theorem 14 *Rabin and Streett games can be solved in time $O(n^{k+1}k!)$ and space $O(n^{k+1}k!)$ where n is the number of states and k is the number of pairs of the winning condition.*

On the one hand, the space complexity of the algorithm makes it prohibitively expensive. Implementing an efficient memory system that supports this algorithm makes it less attractive in practice. On the other hand, if we want to use the intermediate fixpoint values for construction of the winning strategy then memorizing some of the intermediate values is necessary anyway.

7 Conclusions

We show how to define Rabin and Streett ranking, which are a sound and complete way to characterize the winning regions in the respective games. We show that by computing the ranking directly we can solve these games faster. Our algorithms improve the time to solve these kind of games to approximately the square root of previous bounds.

In order to prove completeness of the ranking method, we provide recursive fixpoint algorithms for solving Rabin and Streett games. We then further show that by accelerating the fixpoint computation we get algorithms that match the run time of our explicit algorithm at the price of increasing the space complexity.

Both the enumerative and symbolic algorithms are borrowed from algorithms for solving parity games. This raises the question whether we can adapt the strategy improvement technique [27] as well as other algorithms to solve parity games [1, 15] to Rabin and Streett games. We conjecture that every solution to parity games that works in time $t(m, n, k)$ can be generalized to solve Rabin and Streett games in time $k!t(m, n, 2k)$ (recall that a Rabin / Streett game is converted to a parity game with $2k$ priorities).

We mentioned that the direct rank computation cannot be implemented symbolically. This is similar to Jurdziński's algorithm [14]. Bustan et al. suggested to use Algebraic Decision Diagrams (ADDs) to represent Jurdziński's ranking symbolically [3]. We cannot say whether this would be applicable in our case as well.

References

[1] H. Björklund, S. Sandberg, and S. Vorobyov. A discrete subexponential algorithm for parity games. In *20th STACS*, LNCS 2607, pp. 663–674. Springer-Verlag, 2003.

[2] J. Büchi and L. Landweber. Solving sequential conditions by finite-state strategies. *TAMS*, 138:295–311, 1969.

[3] D. Bustan, O. Kupferman, and M. Vardi. A measured collapse of the modal μ-calculus alternation hierarchy. In *Proc. 21st STACS*, LNCS 2996, pp. 522–533. 2004.

[4] A. Church. Logic, arithmetic and automata. In *Proc. 1962 Int. Congr. Math.*, pages 23–25, Upsala, 1963.

[5] S. Dziembowski, M. Jurdzinski, and I. Walukiewicz. How much memory is needed to win infinite games. In *12th LICS*, pp. 99–110, 1997.
[6] E. Emerson. Automata, tableaux and temporal logics. In *ICLP*, LNCS 193, pp 79–88. Springer-Verlag, 1985.
[7] E. Emerson. Model checking and the μ-calculus. In *Descriptive Complexity and Finite Models*. AMS, 1997.
[8] E. Emerson and C. Jutla. The complexity of tree automata and logic of programs. In *29th FOCS*, pp. 328–337, 1988.
[9] E. Emerson and C. Jutla. Tree automata, μ-calculus and determinacy. In *32nd FOCS*, pp. 368–377, 1991.
[10] E. A. Emerson and C. L. Lei. Efficient model-checking in fragments of the propositional modal μ-calculus. In *1st LICS*, pp. 267–278, 1986.
[11] K. Etessami, T. Wilke, and R. A. Schuller. Fair simulation relations, parity games, and state space reduction for Büchi automata. In *28th ICALP*, LNCS 2076, pp. 694–707. 2001.
[12] Y. Gurevich and L. Harrington. Automata, trees and games. In *14th STOC*, pp. 60–65, 1982.
[13] F. Horn. Streett games on finite graphs. In *2nd GDV*, 2005.
[14] M. Jurdzinski. Small progress measures for solving parity games. In *17th STACS*, LNCS 1770, pp. 290–301. 2000.
[15] M. Jurdziński, M. Paterson, and U. Zwick. A deterministic subexponential algorithm for solving parity games. In *SODA*, 2006.
[16] D. Kozen. Results on the propositional μ-calculus. *TCS*, 27:333–354, 1983.
[17] O. Kupferman and M. Vardi. Weak alternating automata and tree automata emptiness. In *30th STOC*, pp. 224–233, 1998.
[18] O. Kupferman and M. Vardi. Weak alternating automata are not that weak. *TOCL*, 2001(2):408–429, July 2001.
[19] D. Long, A. Brown, E. Clarke, S. Jha, and W. Marrero. An improved algorithm for the evaluation of fixpoint expressions. In *6th CAV*, LNCS 818, pp. 338–350, 1994.
[20] N. Piterman, A. Pnueli, and Y. Sa'ar. Synthesis of Reactive(1) Designs. In *7th VMCAI*, LNCS 3855, pp 364–380, 2006.
[21] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *16th POPL*, pp. 179–190, 1989.
[22] M. Rabin. Decidability of second order theories and automata on infinite trees. *TAMS*, 141:1–35, 1969.
[23] M. Rabin. *Automata on Infinite Objects and Churc's Problem*, volume 13 of *Regional Conference Series in Mathematics*. AMS, 1972.
[24] S. Safra. On the complexity of ω-automata. In *29th FOCS*, pp. 319–327, 1988.
[25] R. Streett. Propositional dynamic logic of looping and converse is elementarily decidable. *IC*, 54:121–141, 1982.
[26] W. Thomas. Automata on infinite objects. In *Handbook of TCS*, volume B, chapter 4, pp. 165–191. MIT Press, 1990.
[27] J. Voge and M. Jurdzinski. A discrete strategy improvement algorithm for solving parity games. In *12th CAV*, LNCS 1855, Springer-Verlag, pp. 202–215, 2000.
[28] I. Walukiewicz. Pushdown processes: Games and model-checking. *IC*, 164(2):234–263, 2001.
[29] T. Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bull. Soc. Math. Belg.*, 8(2), May 2001.