Faster Algorithms for Finitary Games*

Florian Horn

LIAFA, Université Paris 7, Case 7014, 2 place Jussieu, F-75251 Paris 5, France.
e-mail: horn@liafa.jussieu.fr

Abstract. The theory of games is a prominent tool in the controller synthesis problem. ω-regular games offer a clear and robust model of specifications, and present an alternative vision of several logic-related problems. All ω-regular conditions can be expressed by safety and liveness conditions. An issue with the classical definition of liveness specifications is that there is usually no control over the time spent between two occurrences of the desired events. Recently, Chatterjee and Henzinger introduced and studied games based on a finitary notion of liveness, for parity and Streett conditions. We present here faster algorithms for these games, as well as an improved upper bound on the memory needed for the Streett case.

1 Introduction

Games are one of the most practical tools to study the controller synthesis in open systems. The setting of the problem is translated into an arena, while the controller and the environment are the players that makes decisions based on the current state of the system and the actions of their opponent. The desired behaviour of the system is given as a constraint over the sequence of system states, usually an ω-regular condition [MP92]. The study of the resulting ω-regular games is the subject of a very large part of the games theory (for example, [Tho95,AHK02]). Although these games also present the advantage of giving alternate tools to solve problems of model-checking and verification, they present some weakness in the actual synthesis of controllers. ω-regular conditions can be expressed by a liveness part and a safety part. The safety part is sound in terms of controller synthesis: it asks for the controller to prevent the occurrence of a undesirable event, as long as some other condition does not change. Liveness, however, is not so clear. The classical definition asks only for the desired event to happen *eventually*, without any constraints on the number of transitions it may take as long as it is finite. This allows more robust specifications, in the sense that they do not depend on the way

* Work supported by the EU-TMR network GAMES. Some results were found in RWTH, Aachen.

a system is represented, and in one-shot liveness (reachability), this is perfectly natural: the actual number of transitions depends too much on the particular representation we use rather than on the specifics of the system studied. But as soon as we consider Büchi conditions, there exists behaviours compatible with these specifications in which the number of transitions between two visits to the target set is unbounded. On finite graphs, one can always take shortcuts to avoid these cases. But when we consider parity conditions in open systems, there are cases where there is no bounded solution. Finitary conditions, in which the unbounded behaviours are forbidden were introduced in [AH94]. More recently, [BC06] proposed a logic with bounds on the size of states. A fragment of this logic express the finitary conditions.

In [CH06], Chatterjee and Henzinger introduced finitary games and studied the cases of parity and Streett specifications. Their results included determinacy for both games, as well as algorithms computing the winning regions and a study of the memory used by the corresponding strategies. The finitary parity problem was also proved to be in $\mathtt{NP} \cap$ co-$\mathtt{NP}$. We present here faster algorithms for both games, based on Turing reductions to well known games. The finitary parity game problem is proved to be in $\mathtt{P}$, with a time complexity[1] of $c \cdot m \cdot n^2$, whereas the original algorithm of [CH06] had a time complexity of $O(n^{2c-3} \cdot c \cdot n)$. The finitary Streett algorithm is faster than the original reduction to finitary parity ($O(4^k \cdot k^2 \cdot m^2 \cdot n)$ instead of $O((n \cdot k! \cdot k^2)^{2k-3} \cdot m \cdot k! \cdot k^3)$), and yields a strategy for Eve with less memory ($2^k \cdot k^2$ instead of $k!k^2$).

[1] n is the number of states in the graph, m is the number of transitions, c is the number of colors in a parity game, and k is the number of pairs in a Streett game.

2 Definition

A 2-player game is a tuple (V, E, Win) consisting in a graph (E) containing a token, and a winning condition $\mathit{Win} \subseteq V^\omega$. The token is always in one of the states and can only move along the edges. The set of states V is partitioned into Eve's states (V_E, denoted by circles) and Adam's states (V_A, denoted by squares). The owner of the state containing the token chooses the next state. An infinite play $\rho = q_1, q_2, \ldots$ is a sequence of states visited by the token, respecting the edge relation: $(q_i, q_{i+1}) \in E$ for all $i > 0$. We consider only infinite plays, by assuming that every state has at least one successor. A play in Win is winning for Eve. Otherwise, it is winning for Adam. For complexity computations, we will always call n the total number of states, and m the total number of transitions.

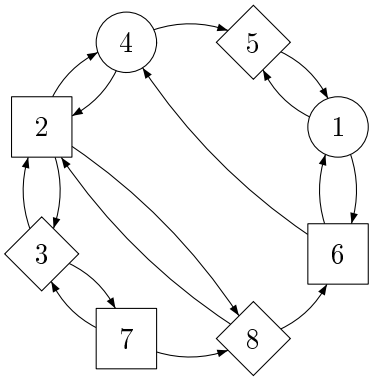


Fig. 1. A game graph

In this paper, we will only consider games on finite graphs. The same notions exists on infinite graphs, but our algorithm are not adapted to them. We will now introduce several notions used to solve games. See [Tho95,Zie98] for more detailed notions and proofs.

Definition 1. *A subgame of a game $G = (V, E, \mathit{Win})$ is a game defined on a subset V' of V such that each state in V' has a successor in V'. The edges and the wining set are restrictions of E and Win to V'.*

The *arena* of a game is the graph (V, E), including the partition between V_A and V_E. A *sub-arena* of an arena is the arena of a subgame. Many notions about games depend only on the arena of the game, and this allows to export these notions from a game to another, as long as they are played on the same arena. The central notion of play, in particular, depends only of the arena.

A strategy for Eve (resp. Adam) is a function σ from V^*V_E (resp. V^*V_A) to V such that for any finite prefix w and any state q, there is an edge between q and $\sigma(w.q)$. Informally, a strategy for player P is a method of extending any finite prefix ending in a state of P. A strategy is positional if σ depends only on the current state. It has a finite memory if it can be realized by a finite-state transducer. A play is consistant with a strategy σ for P if $\forall i \in \mathbb{N}, \rho_i \in V_P \Rightarrow \rho_{i+1} = \sigma(\rho_{1..i})$. All these notions depend only on the arena of the game.

A strategy for P is winning for P if each play consistent with it is won by P. The winning region of a player P in a game G, denoted by $W_P(G)$, is the set of states from where P has a winning strategy.

The *attractor of W for player P in the game G*, denoted $Attr_P^G(W)$, is the set of states from which P can ensure that the token will reach the set W in a finite number of moves. It is computed inductively as usual: $W_0 = W$ and W_{n+1} is the union of W_n, $\{q \in V_P \mid \exists q' \in W_n, (q, q') \in E\}$ and $\{q \in V \mid \forall q' \in W_n, (q, q') \in E\}$. The *attractor strategy* for player P is positional, and consists in always going from a state of W_n to a state in W_{n-1}, thus getting closer to W. The complexity of the computation of either the attractor set or the attractor strategy is $O(m)$.

A *trap* is the dual of an attractor, and hence is a set from which one of the players cannot escape: A trap $T \subseteq V$ for player P is a region such that each state belonging to the other player has a successor in T, and each state in V_P has *all* its successors in T. Note that the complement of an attractor is a trap for the same player, and that a trap is always a sub-arena.

Once again, the notions of attractor and trap depends only on the arena of the game.

In this paper, we will be interested in families of winning conditions that can be played on the same arenas.

3 Parity Games

3.1 Parity Conditions

A parity coloring p is a function that associate an integer to each state of an arena $\mathcal{A}$. A parity arena is an arena equipped with a parity coloring. All the parity games that we define depend only on the parity arena they are played on. It is thus legitimate to talk about "the weak parity game on the arena $\mathcal{A}_p$" without further precision. In complexity computations, we will call c the number of colors

The parity games that are usually studied are the two that follows:

Weak parity games: A play is winning for Eve if the least color appearing in the play is even.

(Classical) parity games: A play is winning if the least color appearing infinitely often in the play is even.

In this paper, we will study another kind of parity games, called finitary parity. These games were introduced by Chatterjee and Henzinger in [CH06]. Intuitively, a play is winning for Eve in finitary parity if for each odd color that occurs infinitely often, a smaller even color occurs

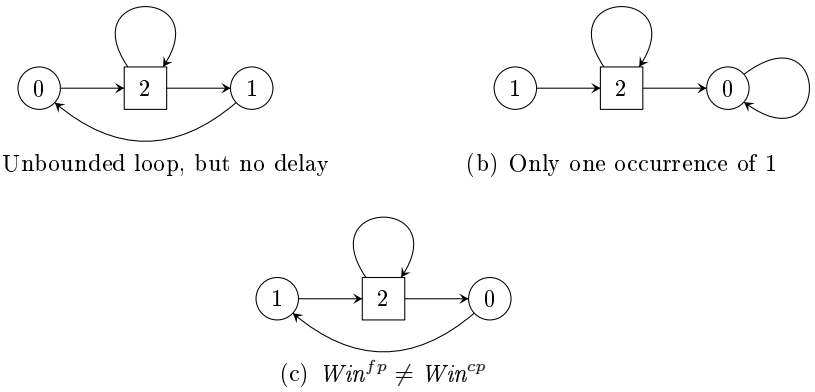
infinitely often, as in classical parity, with the added constraint that the delay between an occurrence of an odd color and the next smaller even color must be ultimately bounded.

The formal definition uses the notion of delay sequence of a play:

Definition 2. *The delay sequence $d(\rho)$ of a play ρ on a parity arena $\mathcal{A}_p$ is defined as follows:*

- *If $p(\rho_i)$ is even, then $d(\rho)_i = 0$.*
- *If $p(\rho_i)$ is odd, then $d(\rho)_i$ is the smallest j such that $p(\rho_{i+j})$ is even and $p((\rho)_{i+j}) < p(\rho_i))$. Note that if there are no such j, $d(\rho)_i = \infty$.*

A play ρ on a parity arena $\mathcal{A}_p$ is winning for Eve in the finitary parity game if and only if $d(\rho)$ is ultimately bounded. Note that, as the delay function can take infinite values, "ultimately bounded" is weaker than simply "bounded".



(a) Unbounded loop, but no delay

(b) Only one occurrence of 1

(c) $Win^{fp} \neq Win^{cp}$

Fig. 2. Example of finitary parity games

Figure 2 gives some examples of how these games work. In the first arena 2(a), Adam can control the time between occurrences of 1, but an occurrence of 0 always comes immediately after. The delay sequence is made only of 0 and 1. Thus Eve wins in the finitary parity game. In the second arena 2(b), Adam can control the time spent between the first 1 and the next 0, or even choose never to go to 0. The first element of the sequence can thus be as high as Adam wants, or even infinite, but all following values will be equal to 0. Thus Eve wins in the finitary parity game. Notice that, in the weak-parity game, Adam would have

won if the play had begun in the state 1. In the third arena 2(c), however, Adam can delay the time between a 1 and the next 0 as long as he wants before allowing the loop to go on. Thus he can make the delay function unbounded and win the finitary game. Notice that he would not win in the classical parity game.

In [CH06], Chatterjee and Henzinger proved the following results about finitary parity games:

- Finitary parity games are determined (i.e. $Win_A^{fp}(\mathcal{A}) \cup Win_E^{fp}(\mathcal{A}) = \mathcal{A}$).
- Eve has a positional strategy.
- Adam has no finite memory strategy.
- Finitary parity games can be solved in time $O(n^{2c-3} \cdot c \cdot n)$.
- Finitary parity games are in NP $\cap$ co-NP.

Our algorithm for finitary games will use yet another kind of parity games, that we called repeating parity game. This games is also defined in terms of delay sequence: A play ρ is winning if the associated delay sequence is bounded. For games on finite arenas, it is easy to see that the bound can be set beforehand at n: If Eve cannot reach a smaller even color in less than n moves, then she cannot reach it at all.

3.2 Algorithms

Another vision of the repeating parity games is to consider them as weak-parity games where Adam can reset the set of visited states whenever he wants. The other constraint, boundedness, is not really a problem for Eve, as we have seen. This intuition is formalized in the lemma 3

Lemma 3. *α : The winning region of Adam in the weak-parity game on an arena $\mathcal{A}_p$ is also winning for him in the repeating parity game on the same arena. The attractor of this region is also winning for him in this game.*

β : If Eve wins everywhere in the weak-parity game on an arena $\mathcal{A}_p$, then she wins everywhere in the repeating parity game on the same arena.

Proof. α : A play is winning for Adam in the weak parity game if the smallest color visited is odd. Obviously, a smaller even color cannot occur later. Thus $Win_A^{wp}(\mathcal{A}) \subseteq Win_A^{rp}(\mathcal{A})$. The case of the attractor is solved by the following observation: If a play ρ is winning for Adam in the repeating parity game, then each play of the form $w.\rho$ is also winning for him in this game. Thus $Attr_A(Win_A^{wp}(\mathcal{A})) \subseteq Win_A^{rp}(\mathcal{A})$ $\square$

β : An arena were Eve wins everywhere in the weak-parity game looks like figure 3.2. In this game, Eve needs only to play according to the attractors' strategies whenever the token is not in one of the top-most regions. This guarantees that in at most n moves, the token will get to an even state in the top-most regions without crossing a smaller odd color. Thus Eve also wins everywhere in the repeating parity game.

□

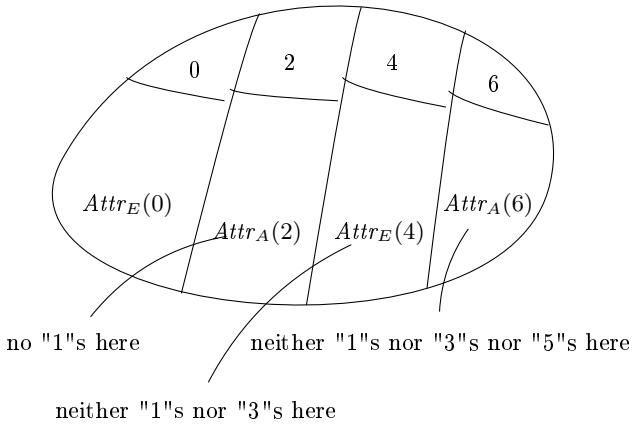


Fig. 3. An arena where Eve wins everywhere in weak-parity

This lemma leads directly to the following algorithm:

Algorithm 1 Algorithm computing the winning regions of Adam and Eve for the repeating parity game

Require: Algorithm for computing the winning regions in a weak-parity game
  input $\mathcal{A}, p$
  $\mathcal{B} \leftarrow \mathcal{A}$
  repeat
    $\mathcal{B} \leftarrow \mathcal{B} \setminus Attr_A(Win_A^{wp}(\mathcal{B}, p))$
  until $Win_A^{wp}(\mathcal{B}, p) = \emptyset$
  return $\mathcal{B}, \mathcal{A} \setminus \mathcal{B}$

The termination of the algorithm 1 is guaranteed by the fact that in each "repeat" loop, $\mathcal{B}$ looses at least one state. This limits the number of such loops to n. As weak-parity games are solved in time $c \cdot m$, the global complexity of our algorithm is $c \cdot m \cdot n$.

The validity of this algorithm comes from lemma 3.

We will now solve finitary games using the same kind of contruction. Lemma 4 gives relations between the winning regions of repeating parity and finitary parity that are very similar to the ones in lemma 3.

Lemma 4. *α : The winning region of Eve in the repeating parity game on an arena $\mathcal{A}_p$ is also winning for her in the finitary parity game on the same arena. The attractor of this region is also winning for her in this game.*

β : If Adam wins everywhere in the repeating parity game on an arena $\mathcal{A}_p$, then he wins everywhere in the finitary parity game on the same arena.

Proof. α : A play is winning for Eve in the repeating parity game if the associated delay function is bounded. It is winning for her in the finitary parity game if this function is ultimately bounded. Thus $Win_E^{rp}(\mathcal{A}_p)) \subseteq Win_E^{fp}(\mathcal{A}_p)$. As the finitary parity condition is prefix independent, we can conclude that $Attr_E(Win_E^{rp}(\mathcal{A}_p)) \subseteq Win_E^{fp}(\mathcal{A}_p)$ □

β : The second part of the proof is more complex. Adam has a strategy π that is winning everywhere in repeating parity. From it, we derive the following strategy π':

1. Set b to 1
2. Play the strategy π with initial memory from the state where the token is now until there is a sequence with an odd priority followed by b moves without seeing a smaller even priority.
3. Increment b
4. Go back to step 2.

It is immediate that if a play consistent with this strategy behave in such a way that Adam goes infinitely often through the loop, then it is winning for Adam in the finitary parity game. The only point that causes trouble is the step 2. But a play that would get stuck in this state would be a play consistent with π where for each occurrence of an odd color, there is an occurrence of a smaller even color in the next b moves. This would be a contradiction to the hypothesis that π is winning for Adam. Thus π' is winning for Adam in the finitary parity game. □

As for repeating parity, we use this lemma to build an algorithm solving finitary parity games.

Algorithm 2 Algorithm computing the winning regions of Adam and Eve for the finitary parity game

Require: Algorithm for computing the winning regions in a repeating parity game
  input $\mathcal{A}, p$
  $\mathcal{B} \leftarrow \mathcal{A}$
  repeat
    $\mathcal{B} \leftarrow \mathcal{B} \setminus Attr_E(Win_E^{rp}(\mathcal{B}, p))$
  until $Win_E^{rp}(\mathcal{B}, p) = \emptyset$
  return $\mathcal{A} \setminus \mathcal{B}, \mathcal{B}$

As in Algorithm 1, the termination and complexity are guaranteed by the fact that the repeat loop removes one state from $\mathcal{B}$. Likewise, the complexity is n times the complexity of the former algorithm, or $c \cdot m \cdot n^2$.

This algorithm is thus much faster than the one of [CH06], and puts the problem in `P` instead of `NP` $\cap$ co-`NP`.

4 Streett Games

4.1 Streett Conditions

A Streett coloring s over an arena $\mathcal{A}$ is a set of pairs of sets of states of $\mathcal{A}$. The first element of a pair is called a "request", and the second element is the corresponding "response". A Streett arena $\mathcal{A}_s$ is an arena equipped with a Streett coloring. The rank of a Streett arena is the number of pairs that constitutes the Streett coloring. The rank of a Streett game is the rank of its arena. In complexity computation, the rank of the Streett condition will be denoted by k. As was the case for parity games, all the variants of Streett games that we will define depends only on the Streett arena they are played on. Again, there are two classical versions of the Streett games:

Weak Streett games: A play is winning for Eve if for each request that occur in the play, the corresponding response also occurs.

(Classical) Streett games: A play is winning for Eve if for each request occurring infinitely often in the play, the corresponding response also occurs infinitely often.

Chatterjee and Henzinger also introduced a finitary version of the Streett games in [CH06]. Intuitively, a play is winning for Eve in finitary Streett if for each request that occurs infinitely often, the corresponding response occurs infinitely often, as in classical Streett, with the added constraint that the delay between an occurrence of a request and the next corresponding response must be ultimately bounded.

The formal definition also uses a notion of delay sequence derived from a play:

Definition 5. *The delay sequence $d(\rho)$ of a play ρ on a Streett arena $\mathcal{A}_p$ is defined as follows:*

- *If ρ_i does not belong to a request, then $d(\rho)_i = 0$.*
- *If ρ_i belong to the request of only one pair, then $d(\rho)_i$ is the smallest j such that ρ_{i+j} belong to the corresponding response. Note that if there are no such j, $d(\rho)_i = \infty$.*
- *If ρ_i belong to several requests, then $d(\rho)_i$ is the maximum of the values computed with the method above for each request.*

A play ρ on a Streett arena $\mathcal{A}_p$ is winning for Eve in the finitary Streett game if and only if $d(\rho)$ is ultimately bounded.

In [CH06], Chatterjee and Henzinger proved the following results about finitary Streett games:

- Finitary Streett games are determined.
- Eve has a strategy that uses $k! \cdot k^2$ memory states.
- Adam has no finite memory strategy.
- Finitary Streett games can be solved in time $O((n \cdot k! \cdot k^2)^{2k-3} \cdot m \cdot k! \cdot k^3)$.

As for parity games, we will use another kind of Streett games in our algorithm, called request-response games. These games were defined and studied by Wallmeier, Thomas and Hutten in [WHT03]. Even if these games do not bear the name Streett, they are defined by a Streett arena:

A play is winning in a request-response game if its delay sequence takes only finite values, i.e. if for each occurrence of a request, there is later an occurrence of a corresponding response.

We will use these games in the finitary Streett algorithm in the same way we used repeating parity games in the finitary parity algorithm. Here, however, there is no link with the weak-Streett games.

[WHT03] presents an algorithm to solve request-response games. It is based on a reduction to generalized Büchi games. The time complexity of their algorithm is $O(4^k \cdot k^2 \cdot m^2)$. The strategy for Eve that derives from this algorithm has the property that in each play consistent with it, each request is matched by a corresponding response in the next $k \cdot n$ moves. This last remark is very important for our purposes, as the definition of request-response games in [WHT03] asks only for finite values of the delay sequence, not necessarily bounded.

There is a lemma that gives the relation between finitary Streett games and request-response games:

Lemma 6. *α : The winning region of Eve in the request-response game on an arena $\mathcal{A}_s$ is also winning for her in the finitary Streett game on the same arena. The attractor of this region is also winning for her in this game.*

β : If Adam wins everywhere in the request-response game on an arena $\mathcal{A}_s$, then he wins everywhere in the finitary Streett game on the same arena.

Proof. α : In the winning region of Eve in the request-response game, she can use the strategy derived from [WHT03]. It guarantees that each request is matched by a corresponding response in the next $k \cdot n$ moves, and thus that the delay sequence is bounded. Thus $Win_E^{rr}(\mathcal{A}_s)) \subseteq Win_E^{fs}(\mathcal{A}_p)$. As the finitary condition is prefix-independent, we get $Attr_E(Win_E^{rr}(\mathcal{A}_s)) \subseteq Win_E^{fs}(\mathcal{A}_p)$. □

β : The construction of a winning strategy for Adam for finitary Streett from a strategy winning everywhere for him in request-response is close to the one used to build a winning strategy for him in finitary parity. If π is a winning strategy for Adam in the request-response game, the strategy π' is defined by:

1. Set b to 1
2. Play the strategy π with initial memory from the state where the token is now until there is a sequence with a request followed by b moves without seeing the corresponding response.
3. Increment b
4. Go back to step 2.

Once again a play that does not get stuck in step 2 is clearly winning for Adam. And a play that would get stuck in the step 2 would be a play where each request is followed by a response, in contradiction with the fact that π is winning in request-response. Thus π' is winning for Adam everywhere in $\mathcal{A}_s$. □

The algorithm obtained from this lemma is :

As in the other algorithms, the number of loops is bounded by the number of states in the arena. The complexity is n times the complexity of the algorithm for request-response games, or $O(4^k \cdot k^2 \cdot m^2 \cdot n)$

Another result that comes from this algorithm concerns the memory that Eve needs to win. Her strategy is made only on attractors, and of the strategies for request-response games, depending only of the position of the token in the arena. The strategy for a request-response game need $k \cdot 2^k$, and the attractor strategy is positional. Thus Eve needs only $k \cdot 2^k$ memory states to win a finitary Streett game on her winning region. This improve

Algorithm 3 Algorithm computing the winning regions of Adam and Eve for the finitary Streett game

Require: Algorithm computing the winning regions in a request-response game
  input $\mathcal{A}, p$
  $\mathcal{B} \leftarrow \mathcal{A}$
  repeat
    $\mathcal{B} \leftarrow \mathcal{B} \setminus Attr_E(Win_E^{rr}(\mathcal{B}, p))$
  until $Win_E^{rr}(\mathcal{B}, p) = \emptyset$
  return $\mathcal{A} \setminus \mathcal{B}, \mathcal{B}$

the result of [CH06], which proved that Eve needed only $k! \cdot k^2$ memory states, from the index of appearance records used in the reduction to finitary parity games ([BLV96]). Interrestingly, the weak-Streett strategies for Eve need less memory, with 2^k([NSW02]), while classical Streett games need $k!$ memory states ([DJW97,Hor05]).

5 Conclusion and Developments

Our algorithms for finitary parity and Streett games are much faster than the version given in the original paper by Chatterjee and Henzinger. The finitary parity problem, in particular, was proved to be in P, improving the former result of NP $\cap$ co-NP. The algorithm for Streett game represents a good improvement in time complexity, and yields more compact strategies for Eve. We had hoped to get a procedure to solve finitary Streett games with a procedure using weak-Streett games, which would have put this problem in PSPACE. However, if there is such a procedure, it eluded us so far.

Our objective in this field of research are an extension of the notion of finitary games to Muller conditions, and the study of links between these games and a fragment of the ωBS-regular logic of Bojanczyk and Colcombet.

References

[AH94] R. Alur and T.A. Henzinger. Finitary Fairness In proceedings of *Logic In Computer Science*, LICS'94, p. 52–61. IEEE Computer Society, 1994.

[AHK02] R. Alur, T.A. Henzinger and O. Kupferman. Alternating-time temporal logic. In *Journal of the ACM*, volume 49, p.672–713. 2002.

[BC06] M. Bojanczyk and T. Colcombet Bounds in ω-regularity In proceedings of *Logic In Computer Science*, LICS'06, p. 285–296, IEEE Computer Society, 2006.

[BLV96] N. Buhrke, H. Lescow and J. Vöge. Strategy Construction in Infinite Games with Streett and Rabin Chain Winning Conditions. In proceedings of *Tools and Algorithms for Construction and Analysis of Systems*, volume 1055 of *Lecture Notes in Computer Science, TACAS'96*, p. 207–224, Springer, 1996.

[CH06] K. Chatterjee and T.A. Henzinger. Finitary Winning in omega-Regular Games. In proceedings of *Tools and Algorithms for the Construction and Analysis of Systems*, volume 3920 of *Lecture Notes in Computer Science, TACAS'06*, p. 257–271, Springer, 2006.

[DJW97] S. Dziembowski, M. Jurdziński and I. Walukiewicz. How Much Memory Is Needed to Win Infinite Games ? In proceedings of *Logic In Computer Science*, LICS'97, p. 99–110, IEEE Computer Society, 1997.

[Jur00] M. Jurdziński Small Progress Measures for Solving Parity Games. In proceedings of *Symposium on Theoretical Aspects of Computer Science*, STACS'00, volume 1770 of *Lecture Notes in Computer Science*, p. 290–301, Springer, 2000

[Hor05] F. Horn. Streett Games on Finite Graphs. *Games in Design and Verification*, Workshop collocated with *Computer Aided Verfication*, 2005

[MP92] Z. Manna and A. Pnueli. The Temporal Logic of Concurrent and Reactive System Springer, 2002.

[NSW02] J. Neumann, A. Szepietowski and I. Walukiewicz. Complexity of weak acceptance conditions in tree automata. In *Information Processing Letters*, volume 84, p181–187, Elsevier, 2002.

[Tho95] W. Thomas. On the Synthesis of Strategies in Infinite Games. In proceedings of *Symposium on Theoretical Aspects of Computer Science*, STACS'95, volume 900 of *Lecture Notes in Computer Science*, p. 1–13, Springer, 1995.

[VJ00] J. Vöge and M. Jurdziński. A Discrete Strategy Improvement Algorithm for Solving Parity Games. In proceedings of *Computer Aided Verfication*, CAV'00, volume 1855 of *Lecture Notes in Computer Science*, p. 202–215, Springer, 2000.

[WHT03] N. Wallmeier, P. Hutten and W. Thomas. Symbolic Synthesis of Finite-State Controllers for Request-Response Specifications. In proceedings of *Conference on Implementation and Application of Automata*, volume 2759 of *Lecture Notes in Computer Science*, p. 11–22, Springer, 2003.

[Zie98] W. Zielonka. Infinite Games on Finitely Coloured Graphs with Applications to Automata on Infinite Trees. In *Theoretical Computer Science*, volume 200(1-2), p. 135–183, 1998