Deciding Parity Games in Quasipolynomial Time*

Cristian S. Calude[1], Sanjay Jain[2], Bakhadyr Khoussainov[1], Wei Li[3] and Frank Stephan[2,3]

[1] Department of Computer Science, University of Auckland
Private Bag 92019, Auckland, New Zealand
{cristian,bmk}@cs.auckland.ac.nz
[2] Department of Computer Science, National University of Singapore
13 Computing Drive, COM1, Singapore 117417, Republic of Singapore
{sanjay,fstephan}@comp.nus.edu.sg
[3] Department of Mathematics, National University of Singapore
10 Lower Kent Ridge Road, S17, Singapore 119076, Republic of Singapore
liwei.sg@gmail.com

Abstract It is shown that the parity game can be solved in quasipolynomial time. The parameterised parity game (with n nodes and m distinct values) is proven to be in the class of fixed parameter tractable (**FPT**) problems (when parameterised over m). Both results improve known bounds, from runtime $n^{O(\sqrt{n})}$ to $O(n^{\log(m)+6})$ (note that $m \leq n$) and from an **XP**-algorithm with runtime $O(n^{\Theta(m)})$ for fixed parameter m to an **FPT**-algorithm with runtime $O(n^5)+g(m)$, for some function g depending on m only. As an application it is proven that coloured Muller games with n nodes and m colours can be decided in time $O((m^m \cdot n)^5)$; this bound cannot be improved to $O((2^m \cdot n)^c)$, for any c, unless **FPT** = **W[1]**.

* S. Jain is supported in part by NUS grant C252-000-087-001; B. Khoussainov is supported in part by the Marsden Fund grant of the Royal Society of New Zealand.

1 Introduction

A parity game is given by a directed graph (V, E), a starting node $s \in V$, a function val which attaches to each $v \in V$ an integer value (also called colour) from a set $\{1, 2, \ldots, m\}$; the main parameter of the game is n, the number of nodes, and the second parameter is m. Two players Anke and Boris move alternately in the graph with Anke moving first. A move from a node v to another node w is valid if (v, w) is an edge in the graph; furthermore, it is required that from every node one can make at least one valid move. Anke and Boris own certain values. The alternate moves by Anke and Boris and Anke and Boris and ... define an infinite sequence of nodes which is called a play. The values of the nodes can always be chosen such that one player owns the even numbers and the other player owns the odd numbers. Anke wins a play through nodes $a_0, a_1, \ldots$ iff the limit superior of the sequence $val(a_0), val(a_1), \ldots$ is a number which she owns, that is, a number of her parity. An example is the game as given in

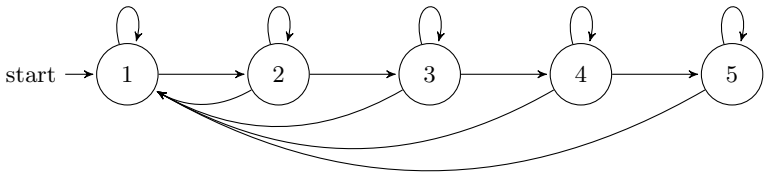

Figure 1.

Figure 1; here the nodes are labeled with their values, which are unique (but this is not obligatory); furthermore, Anke has even and Boris has odd parity. Boris has now the following memoryless (that is, moves are independent of the history) winning strategy for this game: $1 \to 1$, $2 \to 3$, $3 \to 3$, $4 \to 5$, $5 \to 5$. Whenever the play leaves node 1 and Anke moves to node 2, then Boris will move to node 3. In case Anke moves to node 4, Boris will move to node 5. Hence, whenever the play is in a node with even value (this only happens after Anke moved it there), in the next step the play will go into a node with a higher odd value. So the largest infinitely often visited node value is odd and therefore the limit superior of these numbers is an odd number which justifies Boris' win. Hence Boris has a winning strategy for the parity game given above.

For parity games, in general, the winner can always use a *memory-*

less winning strategy [4,19,36,37,46]. This fact will be one central point in the results obtained in this paper: the parity game will be augmented with a special statistics – using polylogarithmic space – which indicates the winner correctly after a finite time whenever the winner employs a memoryless winning strategy.

Parity games are a natural class of games which are not only interesting in their own right, but which are also connected to fundamental notions like μ-calculus, modal logics, tree automata and Muller games [2,5,6,9,18,19,41,43,44,45]. Thus a possible application of good algorithms to solve parity games would be that one could get better algorithms to decide the theory of certain tree automatic structures [15,16,35] and one could employ such algorithms in order to understand these structures better.

For investigating the complexity side of the game, it is assumed that the game is given by a description in size polynomial in the number n of nodes and that one can evaluate all relevant parts of the description in logarithmic space. A possibility is to store the following three items for each game (where Anke moves first and starts from node 1):

- two numbers m, n with $1 \leq m \leq n$ and one bit which says whether the values owned by player Anke are the even or the odd numbers;
- the game graph given by a table, that is, for each pair of nodes, a bit which says whether there is a directed edge between the two nodes or not;
- the values of the nodes given by another table which holds, for each node, a binary number from $\{1, 2, \ldots, m\}$.

An important open problem for parity games is the *time complexity* for finding the winner of a parity game, when both players play optimally; the first algorithms took exponential time [36,46] and subsequent studies searched for better algorithms [29,31,32,39,40,41].

Emerson, Jutla and Sistla [20] showed that the problem is in $\mathbf{NP} \cap \mathbf{co\text{-}NP}$ and Jurdzinski [30] improved this bound to $\mathbf{UP} \cap \mathbf{co\text{-}UP}$. This indicates that the problem is not likely to be hard for exponential time. Indeed, Petersson and Vorobyov [39] devised a subexponential randomised algorithm and Jurdzinski, Paterson and Zwick [32] a deterministic algorithm of similar complexity (more precisely, the subexponential complexity was approximately $n^{O(\sqrt{n})}$). Besides this main result, there are also various practical approaches to solve special cases [2,13,23] or to test out heuristics [26,31]; however, when Friedmann and Lange [21] compared the various parity solving algorithms from the practical side, they found

that Zielonka's recursive algorithm [46] was still the most useful one in practice. Furthermore, McNaughton [36] showed that the winner of a parity game can be determined in time $O(n^m)$ and this was subsequently improved to $O(n^{m/3})$ [40], where n is the number of nodes and m is the maximum value (aka colour aka priority) of the nodes.

The consideration of the parameter m is quite important; Schewe [40,41] argues that for many applications which are solved using parity games, the parameter m is much smaller than n, often by an exponential gap. Also Di Stasio, Murano, Perelli and Vardi [12] investigated in their experiments various scenarios where the number m is logarithmic in n. The present work takes therefore the parameter m into consideration and improves the time bounds bounds in two ways:

- The overall time complexity is $O(n^{\lceil \log(m) \rceil + 6})$ which provides a quasi-polynomial bound on the runtime, as one can always choose $m \leq n$.
- Furthermore, if $m < \log(n)$, then the overall time complexity is $O(n^5)$, which shows that the problem is fixed parameter tractable when parameterised by m; the parity games are therefore in the lowest time complexity class usually considered in parameterised complexity.

Prior investigations had already established that various other parameterisations of parity games are fixed-parameter tractable, but the parameterisation by m was left open until now. An application of the results is that coloured Muller games with n nodes and m colours can be decided in time $O((m^m \cdot n)^5)$; this bound cannot be improved to $O((2^m \cdot n)^c)$ for any c unless $\mathbf{FPT} = \mathbf{W[1]}$.

2 The Complexity of the Parity Game

The main result in this section is an alternating polylogarithmic space algorithm to decide the winner in parity games; later more concrete bounds will be shown. The idea is to collect for both players in the game, Anke and Boris, in polylogarithmic space statistics over their performance in the play: these statistics store information about whether the play has gone through a loop where the largest valued node has the parity of the corresponding player.

The following notation will be used throughout the paper. In order to avoid problems with fractional numbers and $\log(0)$, let $\lceil \log(k) \rceil = \min\{h \in \mathbb{N} : 2^h \geq k\}$. Furthermore, a function (or sequence) f is called *increasing* whenever for all i, j the implication $i \leq j \Rightarrow f(i) \leq f(j)$ holds.

Theorem 1. *There exists an alternating polylogarithmic space algorithm deciding which player has a winning strategy in a given parity game. When the game has n nodes and the values of the nodes are in the set $\{1, 2, \ldots, m\}$, then the algorithm requires $O(\log(n) \cdot \log(m))$ alternating space.*

Proof. The idea of the proof is that, in each play of the parity game, one maintains winning statistics for the players Anke and Boris. These statistics are updated after every move for both players. In case a player plays according to a memoryless winning strategy for the parity game, the winning statistics of this player will eventually indicate the win (in this case one says that the "winning statistics of the player mature") while the opponent's winning statistics will never mature. This will be explained in more detail below.

The winning statistics of Anke (Boris) has the following goal: to track whether the play goes through a cycle where the largest value of a node in the cycle is of Anke's (Boris') parity. Note that if Anke follows a memoryless winning strategy then the play will eventually go through a cycle and the node with the largest value occurring in any cycle the play goes through is always a node of Anke's parity. Otherwise, Boris can repeat a cycle with the largest value being of Boris' parity infinitely often and thus win, contradicting that Anke is using a memoryless winning strategy.

The naive method to do the tracking is to archive the last $2n + 1$ nodes visited, to find two identical moves out of the same node by the same player and to check whose parity has the largest value between these two moves. This would determine the winner in the case that the winner uses a memoryless winning strategy. This tracking needs $O(n \cdot \log(n))$ space – too much space for the intended result. To save space one constructs a winning statistics which still leads to an Anke win in case Anke plays a memoryless winning strategy, but memorises only partial information.

The winning statistics of the players are used to track whether certain sequences of nodes have been visited in the play so far and the largest value of a node visited at the end or after the sequence is recorded. The definitions are similar for both players. For simplicity the definition is given here just for player Anke.

In Anke's winning statistics, an i-sequence is a sequence (not necessarily consecutive, but in order) of nodes $a_1, a_2, \ldots, a_{2^i}$ which has been visited during the play so far such that, for each $k \in \{1, 2, \ldots, 2^i - 1\}$, the maximum value of the nodes visited between a_k and a_{k+1}, that is,

$$\max\{val(a) : a = a_k \vee a = a_{k+1} \vee a \text{ was visited between } a_k \text{ and } a_{k+1}\},$$

is of Anke's parity. The aim of Anke is to find a sequence of length at least $2n+1$, as such a sequence must contain a loop. So she aims for an $(\lceil \log(n) \rceil + 2)$-sequence to occur in her winning statistics. Such a sequence is built by combining smaller sequences over time in the winning statistics.

Here a winning statistics of a player consists of $(b_0, b_1, \ldots, b_{\lceil \log(n) \rceil + 2})$ where $b_i = 0$ indicates that currently no i-sequence is being tracked and $b_i > 0$ indicates that an i-sequence is being tracked and that the largest value of a node visited at the end or after this i-sequence is b_i. Note that for each i at most one i-sequence is tracked. The value b_i is the only information of an i-sequence which is kept in the winning statistics.

The following invariants are kept throughout the play and are formulated for Anke's winning statistics; those for Boris' winning statistics are defined with the names of Anke and Boris interchanged. In the below description, "i-sequence" always refers to the i-sequence being tracked in the winning statistics.

- Only b_i with $0 \leq i \leq \lceil \log(n) \rceil + 2$ are considered and each such b_i is either zero or a value of a node which occurs in the play so far.
- An entry b_i refers to an i-sequence which occurred in the play so far iff $b_i > 0$.
- If b_i, b_j are both non-zero and $i < j$ then $b_i \leq b_j$.
- If b_i, b_j are both non-zero and $i < j$, then in the play of the game, the i-sequence starts only after a node with value b_j was visited at or after the end of the j-sequence.

When a play starts, the winning statistics for both players are initialised with $b_i = 0$ for all i. During the play when a player moves to a node with value b, the winning statistics of Anke is updated as follows – the same algorithm is used for Boris with the names of the players interchanged everywhere.

- If b is of Anke's parity or $b > b_i > 0$ for some i, then one selects the largest i such that
  (a) either b_i is not of Anke's parity – that is, it is either 0 or of Boris's parity – but all b_j with $j < i$ and also b are of Anke's parity
  (b) or $0 < b_i < b$

  and then one updates $b_i = b$ and $b_j = 0$ for all $j < i$.
- If this update produces a non-zero b_i for any i with $2^i > 2n$ then the play terminates with Anke being declared winner.

Example. Here is an example of i-sequences for player Anke. This example is only for illustrating how the i-sequences and b_i's work; in particular this example does not use memoryless strategy for any of the players. Consider a game where there is an edge from every node to every node (including itself) and the nodes are $\{1, 2, \ldots, 7\}$ and have the same values as names; Anke has odd parity. Consider the following initial part of a play:

```
1 6 7 5 1 4 5 3 2 1 3 2 3 1 3 3 1 2 1
```

The i-sequences and the b_i's change over the course of above play as given in the following table. In the table, the nodes prefixed by "i :" are the nodes of an i-sequence. For i-sequences, if the updates (a) and (b) apply to the same level i then (a) has priority – for the b_i this does not make a difference and therefore this rule is not part of the updating algorithm.

Move	b_4, b_3, b_2, b_1, b_0	i-sequences in play so far
1	0,0,0,0,1	0:1
6	0,0,0,0,6	0:1 6
7	0,0,0,0,7	1 6 0:7
5	0,0,0,5,0	1 6 1:7 1:5
1	0,0,0,5,1	1 6 1:7 1:5 0:1
4	0,0,0,5,4	1 6 1:7 1:5 0:1 4
5	0,0,0,5,5	1 6 1:7 1:5 1 4 0:5
3	0,0,3,0,0	1 6 2:7 2:5 1 4 2:5 2:3
2	0,0,3,0,0	1 6 2:7 2:5 1 4 2:5 2:3 2
1	0,0,3,0,1	1 6 2:7 2:5 1 4 2:5 2:3 2 0:1
3	0,0,3,3,0	1 6 2:7 2:5 1 4 2:5 2:3 2 1:1 1:3
2	0,0,3,3,0	1 6 2:7 2:5 1 4 2:5 2:3 2 1:1 1:3 2
3	0,0,3,3,3	1 6 2:7 2:5 1 4 2:5 2:3 2 1:1 1:3 2 0:3
1	0,1,0,0,0	1 6 3:7 3:5 1 4 3:5 3:3 2 3:1 3:3 2 3:3 3:1
3	0,3,0,0,0	1 6 3:7 3:5 1 4 3:5 3:3 2 3:1 3:3 2 3:3 3:1 3
3	0,3,0,0,3	1 6 3:7 3:5 1 4 3:5 3:3 2 3:1 3:3 2 3:3 3:1 3 1:3
1	0,3,0,1,0	1 6 3:7 3:5 1 4 3:5 3:3 2 3:1 3:3 2 3:3 3:1 3 1:3 1:1
2	0,3,0,2,0	1 6 3:7 3:5 1 4 3:5 3:3 2 3:1 3:3 2 3:3 3:1 3 1:3 1:1 2
1	0,3,0,2,1	1 6 3:7 3:5 1 4 3:5 3:3 2 3:1 3:3 2 3:3 3:1 3 1:3 1:1 2 0:1

The 3-sequence has already a loop, as there are three occurrences of "3 : 3" and the second and third of these have that the same player moves. However, as the sequences are not stored but only the b_i, Anke's winning statistics only surely indicates a win of player Anke when there is an $i > \log(2n + 1)$ with $b_i > 0$; this i is 4 as $2^4 > 2 \cdot 7 + 1$.

Verification of the algorithm. Note that, at the update of Anke's winning statistics, if b is of Anke's parity, then there is an i that satisfies (a), as otherwise the algorithm would have terminated earlier. Also, the

invariants listed above are preserved after each update by the algorithm, where the i-sequences may be modified after each update as below.

When updating Anke's winning statistics by case (a), one forms a new i-sequence of length 2^i by putting the older j-sequences for $j = i-1, i-2, \ldots, 1, 0$ together and appending the newly visited one-node sequence with value b; when $i = 0$, one forms a new 0-sequence of length 2^0 consisting of just the newly visited node with value b. Note that in case $i > 0$ both b and b_0 are of Anke's parity and therefore the highest valued node between the last member a of the older 0-sequence and the last node in the new i-sequence (both inclusive) has the value $\max\{b_0, b\}$. Furthermore, for every $j < i-1$, for the last node a of the older $(j+1)$-sequence and the first node a' of the older j-sequence, in the new i-sequence a highest valued node in the play between these two nodes a, a' (both inclusive) has value b_{j+1} which, by choice, has Anke's parity. Thus the overall combined new sequence indeed satisfies the properties needed for an i-sequence, and b is the value of the last node of this sequence and thus, currently, also the largest valued node at or after the end of the sequence. All older j-sequences with $j < i$ are discarded and thus their entries are set back to $b_j = 0$.

The same rules apply to the updates of Boris' winning statistics with the roles of Anke and Boris interchanged everywhere.

Claim 2. *If a player is declared a winner by the algorithm, then the play contains a loop with its maximum valued node being a node of the player.*

To prove the claim, it is assumed without loss of generality that Anke is declared the winner by the algorithm. The play is won by an i-sequence being observed in Anke's winning statistics with $2^i > 2n$; thus some node occurs at least three times in the i-sequence and there are $h, \ell \in \{1, 2, \ldots, 2^i\}$ with $h < \ell$ such that the same player moves at a_h and a_ℓ and furthermore $a_h = a_\ell$ with respect to the nodes $a_1, a_2, \ldots, a_{2^i}$ of the observed i-sequence. The maximum value b' between a_h and a_ℓ in the play is occurring between some a_k and a_{k+1} (both inclusive) for a k with $h \leq k < \ell$. Now, by the definition of an i-sequence, b' has Anke's parity. Thus a loop has been observed for which the maximum value of a node in the loop has Anke's parity.

Claim 3. *If a player follows a memoryless winning strategy, then the opponent is never declared a winner.*

To prove the claim, suppose that a player follows a memoryless winning strategy but the opponent is declared a winner. Then the opponent, by

Claim 2, goes into a loop with the maximum node of the opponent's parity. Hence, the opponent can cycle in that loop forever and win the play, a contradiction.

Claim 4. *If a player follows a memoryless winning strategy then the player is eventually declared a winner.*

To prove the claim, it is assumed that the player is Anke, as the case of Boris is symmetric. The values b_i analysed below refer to Anke's winning statistics.

Assume that an infinite play of the game has the limit superior c which, by assumption, is a value of Anke's parity. Consider a step as making a move and updating of the statistics. For each step t let

$$card(c,t) = \sum_{\{k:\ b_k(t) \text{ has Anke's parity and } b_k(t) \geq c\}} 2^k$$

where the $b_k(t)$ refer to the value of b_k at the end of step t (that is, after the updates in the statistics following the t-th move in the play). Now it is shown that whenever at steps t, t' with $t < t'$, a move to node with value c was made and no move, strictly between steps t, t', was made to any node with value $c' \geq c$, then $card(c,t) < card(c,t')$. To see this, let i be the largest index for which there is a step t'' with $t < t'' \leq t'$ such that b_i is updated at step t''. Next one considers all possible cases:

- Case $b_i(t'') = 0$: This case only occurs if b_{i+1} also gets updated at step t'' and contradicts the choice of i. Thus this case does not need to be considered.
- Case $b_i(t) \geq c$ and $b_i(t)$ has Anke's parity: In this case, the only way to update b_i at step t'' is to do an update of type (a). In this case also the entry $b_{i+1}(t'')$ will be updated, in contradiction with the choice of i. Thus this case also does not need to be considered.
- Case $b_i(t) \geq c$ and $b_i(t)$ has Boris' parity: An update at step t'' is possible only by case (a). If $b_i(t'') < c$ then, at step t', another update will occur and will enforce by (b) that $b_i(t') = c$. The value $card(c,t)$ is largest when all $b_j(t)$ with $j < i$ have Anke's parity at step t; even in this worst case, the inequality $card(c,t') - card(c,t) \geq 2^i - \sum_{j:j<i} 2^j \geq 1$ holds.
- Case $0 < b_i(t) < c$: Later at stage t', as an update of type (b) at i is possible, it will be enforced that $b_i(t') = c$, while $b_j(t) < c$ for all $j \leq i$; therefore $card(c,t') \geq card(c,t) + 2^i \geq card(c,t) + 1$.

– Case $b_i(t) = 0$: At stage t'' an update of type (a) will make $b_i(t'') > 0$ and, in case that $b_i(t'') < c$, a further update of type (b) at stage t' will enforce that $b_i(t') = c$. Again, the value $card(c, t)$ is largest when all $b_j(t)$ with $j < i$ have Anke's parity at step t; even in this worst case the inequality $card(c, t') - card(c, t) \geq 2^i - \sum_{j:j<i} 2^j \geq 1$ holds.

Accordingly, once all moves involving nodes larger than c in value have been done in the play, there will still be infinitely many moves to nodes of value c and for each two subsequent such moves at t, t' the inequality $card(c, t) + 1 \leq card(c, t')$ will hold. Consequently, the number $card(c, t)$, for sufficiently large t where a move to a node with value c is made at step t, rely on some i with $b_i(t) \geq c$ and $2^i > 2n$ and later the termination condition of Anke will terminate the play with a win.

The above arguments show that a deterministic alternating Turing machine can simulate both players and, taking the winning statistics into account, will accept the computation whenever Anke has a winning strategy for the game. Moreover, the deterministic alternating Turing machine uses only $O(\log(n) \cdot \log(m))$ space to decide whether the parity game, from some given starting point, will be won by Anke (or Boris), provided the winner plays a memoryless winning strategy (which always exists when the player can win the parity game). □

Chandra, Kozen and Stockmeyer [7] showed how to simulate a deterministic alternating Turing machine working in polylogarithmic space by a deterministic Turing machine working in quasipolynomial time. Their simulation bounds for the alternating Turing machine described in Theorem 1 give a deterministic Turing machine working in time $O(n^{c \log(m)})$ for some constant c. As mentioned above, one can always assume that in a parity game with n nodes, with values from $\{1, 2, \ldots, m\}$, one can choose $m \leq n$, so one gets the following parameterised version of the main results that parity games can be solved in quasipolynomial time.

Theorem 5. *There is an algorithm which decides in time $O(n^{c \log(m)})$ which player has a winning strategy in a parity game with n nodes and values from $\{1, 2, \ldots, m\}$, with $m \leq n$.*

For some special choices of m with respect to n, one can obtain even a polynomial time bound. McNaughton [36] showed that for every constant m, one can solve a parity game with n nodes having values from $\{1, 2, \ldots, m\}$ in time polynomial in n; however, in all prior works the degree of this polynomial depends on m [22]; subsequent improvements were

made to bring the dependence from approximately n^m down to approximately $n^{m/3}$ [31,40]. The following theorem shows that one can bound the computation time by a fixed-degree polynomial in n, for all pairs (m, n) with $m < \log(n)$.

Theorem 6. *If $m < \log(n)$ then one can solve the parity game with n nodes having values from $\{1, 2, \ldots, m\}$ in time $O(n^5)$.*

Proof. Note that Theorem 1 actually shows that the following conditions are equivalent:

- Anke can win the parity game.
- Anke can play the parity game such that her winning statistics matures while Boris' winning statistics does not mature.

Thus one can simplify the second condition and show that it is equivalent to the following two games [34,42]:

- One only maintains Anke's winning statistics and a play terminates with a win for Anke iff she is eventually declared a winner and the game is a win for Boris iff it runs forever;
- One only maintains Boris' winning statistics and a play is a win for Anke iff it never happens that the winning statistics of Boris make him to be declared a winner.

The first game is called a reachability game [34] and the second game a survival game [42, Chapter 9]. Both games are isomorphic, as they are obtained from each other only by switching the player who is supposed to win. Such type of reductions, though not with good complexity bounds, were also considered by Bernet, Janin and Walukiewicz [1]. The reachability game to which one reduces the parity game, can now be described as follows.

- The set Q of nodes of the reachability game consists of nodes of the form $(a, p, \tilde{b})$ where a is a node of the parity game, the player $p \in \{\text{Anke}, \text{Boris}\}$ moves next and $\tilde{b}$ represents the winning statistics of Anke.
- The starting node is $(s, p, \tilde{0})$, where $\tilde{0}$ is the vector of all b_i with value 0, s is the starting node of the parity game and p is the player who moves first.
- Anke can move from $(a, \text{Anke}, \tilde{b})$ to $(a', \text{Boris}, \tilde{b}')$ iff she can move from a to a' in the parity game and this move causes Anke's winning statistics to be updated from $\tilde{b}$ to $\tilde{b}'$ and $\tilde{b}$ does not yet indicate a win of Anke.

– Boris can move from $(a, \text{Boris}, \tilde{b})$ to $(a', \text{Anke}, \tilde{b}')$ iff he can move from a to a' in the parity game and this move causes Anke's winning statistics to be updated from $\tilde{b}$ to $\tilde{b}'$ and $\tilde{b}$ does not yet indicate a win of Anke.

The number of elements of Q can be bounded by $O(n^4)$. First note that the number of increasing functions from $\{0, 1, \ldots, \lceil \log(n) \rceil + 2\}$ to $\{1, 2, \ldots, \lceil \log(n) \rceil\}$ can be bounded by $O(n^2)$, as any such sequence $(b'_0, b'_1, \ldots, b'_{\lceil \log(n) \rceil + 2})$ can be represented by the subset $\{b'_k + k : 0 \leq k \leq \lceil \log(n) \rceil + 2\}$ of $\{1, 2, \ldots, 2\lceil \log(n) \rceil + 2\}$ and that there are at most $O(n^2)$ such sets. Further, note that $b'_k \leq b'_{k+1}$ implies $b'_k + k < b'_{k+1} + k + 1$ and thus all b'_k can be reconstructed from the set. Given a winning statistics $\tilde{b} = (b_0, b_1, \ldots, b_{\lceil \log(n) \rceil + 2})$, one defines $b'_0 = \max\{1, b_0\}$ and $b'_{k+1} = \max\{b'_k, b_{k+1}\}$ and notes that only those b_k with $b_k = 0$ differ from b'_k. Thus one needs $\lceil \log(n) \rceil + 3$ additional bits to indicate which b_k is 0. The overall winning statistics can then be represented by $3\lceil \log(n) \rceil + 5$ bits. Furthermore, one needs 1 bit to represent the player and $\lceil \log(n) \rceil$ bits to represent the current node in the play. Accordingly, each node in Q can be represented with $4\lceil \log(n) \rceil + 6$ bits resulting in $O(n^4)$ nodes in Q. The set Q itself can be represented by using a set of such representations of nodes.

The reachability game can be decided in time $O(|Q| \cdot n)$ by a well-known algorithm. For the general case of a reachability game, the time complexity is linear in the number of vertices plus number of edges of the game graph. The algorithm is listed explicitly by Khaliq and Imran [33] and in a slightly modified version in Grädel's tutorial slides [24,25]. The algorithm is now included for the reader's convenience.

First, one can construct the set Q of vertices and determine a list of nodes $Q' \subseteq Q$ where Anke's winning statistics indicate a win in time $O(|Q| \cdot n)$; the set Q' is the set of target nodes in the reachability game.

Second, one computes for each node $q \in Q$, a linked list of q's successors (which are at most n in number) and a linked list of q's predecessors. Note that the collection of all the successor and predecessor lists for different nodes in Q taken together has the length at most $|Q| \cdot n$. These lists can also be generated in time $O(|Q| \cdot n)$.

Note that a node q is a winning node for Anke if $q \in Q'$ or either Anke moves from q and one successor node of q is a winning node for Anke or Boris moves from q and all successor nodes of q are winning node for Anke. This idea leads to the algorithm below.

Next, for each node q, a tracking number k_q is introduced and maintained such that the winning nodes for Anke will eventually all have $k_q = 0$, where k_q indicates how many further times one has to visit the

node until it can be declared a winning node for Anke. The numbers k_q are initialised by the following rule:

- On nodes $q \in Q'$ the number k_q is 1;
- On nodes $q = (a, \text{Anke}, \tilde{b}) \notin Q'$, the number k_q is initialised as 1;
- On nodes $q = (a, \text{Boris}, \tilde{b}) \notin Q'$, the number k_q is initialised as the number of nodes q' such that Boris can move from q to q'.

These numbers can be computed from the length of the list of successors of q, for each $q \in Q$. Now one calls the following recursive procedure, initially for all $q \in Q'$ such that each call updates the number k_q. The recursive call does the following:

- If $k_q = 0$ then return without any further action else update $k_q = k_q - 1$.
- If after this update it still holds $k_q > 0$, then return without further action.
- Otherwise, that is when k_q originally was 1 when entering the call, recursively call all predecessors q' of q with the same recursive call.

After the termination of all these recursive calls, one looks at k_q for the start node q of the reachability game. If $k_q = 0$ then Anke wins else Boris wins.

In the above algorithm, the predecessors of each node $q \in Q$ are only called at most once, namely when k_q goes down from 1 to 0; furthermore, this is the time where it is determined that the node is a winning node for Anke. Thus there are at most $O(|Q| \cdot n)$ recursive calls and the overall complexity is $O(|Q| \cdot n)$.

For the verification, the main invariant is that, for nodes $q \in Q - Q'$, k_q indicates how many more successors of q one still has to find which are winning nodes for Anke until q can be declared a winning node for Anke. In case that Anke's winning statistics has matured in the node q, the value k_q is taken to be 1 so that the node is processed once in all the recursive calls in the algorithm. For nodes where it is Anke's turn to move, only one outgoing move which produces a win for Anke is needed. Consequently, one initialises k_q to 1 and as soon as this outgoing node is found, k_q goes to 0, which means that the node is declared a winning node for Anke. In case the node q is a node where Boris moves then one has to enforce that Boris has no choice but to go to a winning node for Anke. Thus k_q is initialised to the number of moves which Boris can make in this node; each time when one of these successor nodes is declared a winning

node for Anke, k_q goes down by one. Observe that once the algorithm is completed, the nodes with $k_q = 0$ are exactly the winning nodes for Anke in the reachability game. $\square$

This special case shows that, for each constant m, the parity game with n nodes having values from $\{1, 2, \ldots, m\}$ can be solved in time $O(n^5)+g(m)$, for some function g depending only on m, and the constant in $O(n^5)$ being independent of m. Such problems are called fixed parameter tractable (**FPT**), as for each fixed parameter m the corresponding algorithm runs in polynomial time and this polynomial is the same for all m, except for the additive constant $g(m)$ depending only on m. See Downey and Fellows [14] for an introduction to parameterised complexity.

The next result carries over the methods of the previous result to the general case: the size of the code representing a winning strategy for Boris is given by $\lceil \log(n) \rceil + 3 \leq \log(n) + 4$ numbers of $\lceil \log(m+1) \rceil \leq \log(m) + 1$ bits. As $\log(m) \leq \log(n)$, the overall size of representing a node in the set Q of nodes of the reachability game can be estimated to $\log(n) \cdot (\log(m) + 5) + c$. Hence, the size of $|Q|$ is $O(n^{\log(m)+5})$.

One can combine this with the usual repeated tests for various types of **NP**-problems: for finding the winning strategy for a player which has a winning strategy, say Anke, in the parity game on graph (V, E), one keeps doing the following:

1. One maintains, for each node $a \in V$, a list of possible successors V_a which is initialised as $\{b : (a, b) \in E\}$ at the beginning.
2. If there is no node $a \in V$ with, currently, $|V_a| > 1$,
   then one terminates with a winning strategy for Anke in the parity game being to move from every node a to the unique node in V_a,
   else one selects a node $a \in V$ with $|V_a| > 1$.
3. Now one splits V_a into two nearly equal sized subsets V'_a and V''_a with $|V'_a| \leq |V''_a| \leq |V'_a| + 1$.
4. One replaces V_a by V'_a and permits, in the derived reachability game, moves from $(a, Anke, \tilde{b})$ to $(a', Boris, \tilde{b}')$ only when $a' \in V_a$.
5. If Anke does not win this game, then one replaces $V_a = V''_a$; else one keeps $V_a = V'_a$.
6. Go to step 2.

The above algorithm works since whenever Anke has a winning strategy for the parity game, then there is a memoryless one and therefore when splitting the options at node a, some memoryless winning strategy either always takes a node from V'_a or always takes a node from V''_a. It is straightforward to verify that the above loop runs $n \log(n)$ rounds and each round

involves $O(|Q| \cdot n)$ time plus one solving of the reachability game, which can also be solved in time $O(|Q| \cdot n)$. Thus one can derive the following result.

Theorem 7. *There is an algorithm which decides in time $O(n^{\log(m)+6})$ which player has a winning strategy for a parity game with n nodes and values from $\{1, 2, \ldots, m\}$. Furthermore, the algorithm can compute a memoryless winning strategy of the winner in time $O(n^{\log(m)+7} \cdot \log(n))$.*

3 Parity Games versus Muller Games

Muller games are a well-studied topic [5,6,36,44,46] and they had been investigated as a general case already before researchers aimed for the more specific parity games. A Muller game (V, E, s, G) consists of a directed graph (V, E), a starting node s and a set $G \subseteq \{0, 1\}^V$. For every infinite play starting in s, one determines the set U of nodes visited infinitely often during the play: if $U \in G$ then Anke wins the play else Boris wins the play. In a Muller game the complement of G is closed under union iff for all $U, U' \notin G$, the set $(U \cup U')$ is not in G.

For complexity assumptions, it is natural to consider the case where G is not given as an explicit list, but as a polynomially sized polynomial time algorithm computing the membership of a set U (given by its explicit list) in the set G or some similar equivalent effective representation. The reason for considering such a representation for G is that Horn [27] showed that if G is given as an explicit list of all possible sets of nodes infinitely visited when Anke wins, then the resulting game is solvable in polynomial time in the sum of the number of nodes and the number of explicitly listed sets. Hence, only more flexible ways of formulating winning conditions permit to cover interesting cases of Muller games.

For Muller games, Björklund, Sandberg and Vorobyov [3] considered a parameter which is given by the number of colours. For this, they assign to every node a value or colour from $\{1, 2, \ldots, m\}$ and take G to be some set of subsets of $\{1, 2, \ldots, m\}$. Then U is not the set of infinitely often visited nodes, but instead, the set of colours of the infinitely often visited nodes. Again, if $U \in G$, then Anke wins the play, else Boris wins the play. Coloured Muller games permit more compact representations of the winning conditions. In the worst case there is a 2^m-bit vector, where m is the number of colours; however, one also considers the case where this compressed winning condition is given in a more compact form, say by a polynomial sized algorithm or formula.

In the following, the interactions between Muller games, memoryless

winning strategies and parity games are presented. The proofs of the first two results are only given for the sake of a self-contained paper. The first result is due to Zielonka [46, Corollary 11] and the second one is proven using standard methods.

Theorem 8 (Zielonka [46]). *Assume that in a Muller game (V, E, s, G) the complement of the set G of winning conditions is closed under union. Now, if Anke has a winning strategy in this Muller game then Anke has also a memoryless winning strategy.*

Proof. The possible choices for Anke at any node will be progressively constrained. The proof is by induction on the number of possible moves of Anke in the constrained game. The result holds when, for each node, Anke has only one choice of move. For the induction step, suppose some node v for Anke's move has more than one choice. It is now shown that for some fixed Anke's move at node v, Anke has a winning strategy; thus one can constrain the move of Anke at node v and by induction we are done. Suppose, by way of contradiction, that for every Anke's move w at v, Boris has a winning strategy S_w. This allows Boris to have a winning strategy for the whole game as follows.

Assume without loss of generality that the play starts with Anke's move at v. Intuitively, think of Boris playing several parallel plays against Anke (each play in which Anke moves w at node v, for different values of w) which are interleaved. For ease of notation, consider the individual play with Anke using move w at node v as play H_w, and the interleaved full play as H.

Initially H and all the plays H_w, are at the starting point. At any time in the play H, if it is Anke's move at v and Anke makes the move w', then Boris continues as if it is playing the play $H_{w'}$ (and suspends the previous play H_w if $w \neq w'$). Thus the nodes visited in H can be seen as the merger of the nodes visited in the plays H_w, for each choice w of Anke at node v. This implies that the set of nodes visited infinitely often in H is equal to the union of the sets of nodes visited infinitely often in the various H_w. As Boris wins each play H_w which is played for infinitely many moves, by closure of the complement of G under union, Boris wins the play H. □

As a parity game is also a Muller game in which G is closed under union for both Anke and Boris, the following corollary holds.

Corollary 9 (Emerson and Jutla [19], Mostowski [37]). *The winners in parity games have memoryless winning strategies.*

Hunter [28, page 23] observed the following result. Note that McNaughton also investigated Muller games with memoryless strategies and characterized them through the concept of splitting [36], which is just another way of stating that both G and its complement are union-closed. However, his paper does not connect these Muller games with parity games explicitly.

Theorem 10 (Hunter [28]). *Every Muller game (V, E, s, G) in which both G and its complement are closed under the union operation is a parity game.*

Proof. In this proof a parity game isomorphic to the given Muller game will be constructed. In this parity game player Anke owns the nodes with even value and Boris owns the nodes with odd value. Given V, let

$$V_1 = \{a \in V : \{a\} \in G\} \text{ and } V_2 = \{b \in V : \{b\} \notin G\}.$$

Obviously V is the disjoint union of V_1 and V_2. By the closure under union, any subset $V' \subseteq V_1$ is in G and no subset $V' \subseteq V_2$ is in G.

To prove the theorem, values will be inductively assigned to the nodes one by one.

Suppose values have already been assigned to all nodes in $V - V'$, where V' is initially V. Then, assign the value to one node in V' as follows. Let $V'_1 = V' \cap V_1$ and $V'_2 = V' \cap V_2$.

Case 1: Suppose $V' \in G$. Then, there is a node $a \in V'_1$ such that $\{a\} \cup V'_2 \in G$; otherwise, $V' \notin G$ since complement of G is closed under the union operation. Now let $V''_1 \subseteq V'_1$ and $V''_2 \subseteq V'_2$. The set $\{a\} \cup V''_2$ is in G, as otherwise $(\{a\} \cup V''_2) \cup V'_2$ is not in G, in contradiction to the choice of a. Furthermore, as $V''_1 \cup \{a\} \in G$, $(V''_1 \cup \{a\}) \cup (\{a\} \cup V''_2) = \{a\} \cup V''_1 \cup V''_2$ is in G. Thus whenever $V'' \subseteq V'$ and $a \in V''$, $V'' \in G$. Hence, the value $2|V'|$ is assigned to a accordingly.

Case 2: Suppose $V' \notin G$. Then, there exists a node $b \in V'_2$ such that $\{b\} \cup V'_1 \notin G$, by reasons similar to those given in Case 1. Note that this implies that whenever $V'' \subseteq V'$ and $b \in V''$ then $V'' \notin G$. Hence, the value $2|V'| + 1$ is assigned to b.

The above process of assigning values to nodes is clearly consistent, as in Case 1, if a is in V'' then Anke wins and in Case 2, if b is in V'' then Boris wins. It follows that this Muller game is a parity game. □

Remark 11. For the above proof, it should be noted that the complexity of the reduction is polynomial time, whenever G can be decided in polynomial time.

Besides the coloured Muller game of Björklund, Sandberg and Vorobyov [3], one can also consider the memoryless coloured Muller game. These are considered in order to see whether the game is easier to solve if one permits Anke only to win when she follows a memoryless strategy, otherwise she looses by the rules of the game. The main finding is that while memoryless coloured Muller games are, on one hand, easier in terms of the complexity class to which they belong, and, on the other hand, their time complexity is worse and one cannot exploit small numbers of colours as well as in Muller games. This is the main message of the following lines.

Björklund, Sandberg and Vorobyov [3] proved that the coloured Muller game is fixed-parameter tractable iff the parity game is fixed-parameter tractable (with respect to the number of values m of the parity game). It follows from Theorem 6 that also the coloured Muller game is fixed-parameter tractable.

More precisely, Björklund, Sandberg and Vorobyov [3] showed that a coloured Muller game with m colours and n nodes can be translated into a parity game with $2m$ colours and $m! \cdot n$ nodes. Note that $\log(m! \cdot n) \geq 2m$ for all $m \geq 24$ and $n \geq 1$: $\log(m!) \geq \log(8^{m-8}) \geq 3 \cdot (m-8) = 3m - 24$. For $m \geq 24$, $3m - 24 \geq 2m$. Furthermore, for all sufficiently large n, $\log(m! \cdot n) \geq 2m$. Thus, for almost all pairs of (m, n), $\log(m! \cdot n) \geq 2m$ and therefore one can use the polynomial time algorithm of Theorem 6 to get the following explicit bounds.

Theorem 12. *One can decide in time $O(m^{5m} \cdot n^5)$ which player has a winning strategy in a coloured Muller game with m colours and n nodes.*

For the special case of $m = \log(n)$, the corresponding number of nodes in the translated parity game is approximately $n^{\log(\log(n))+2}$ and the polynomial time algorithm of Theorem 6 becomes an $O(n^{5 \log \log(n)+10})$ algorithm. The algorithm is good for this special case, but the problem is in general hard and the algorithm is slow.

One might ask whether this bound can be improved. Björklund, Sandberg and Vorobyov [3] showed that under the Exponential Time Hypothesis (which says that the problem **3SAT** with n variables is not solvable in time $O(2^{o(n)})$) it is impossible to improve the above algorithm to $O(2^{o(m)} \cdot n^c)$, for any constant c. The following result enables to get a slightly better lower bound based on the more likely assumption that **FPT** $\neq$ **W[1]**. Here a dominating set of a graph is a set of nodes such that from every node in the graph there is an edge to one of the nodes in the dominating set; for this property one deviates from the usual conven-

tion of the non-existence of self-loops and assumes that every node has a loop to itself.

Theorem 13. *Given a graph H with n nodes and a number m with $1 \leq m \leq n$, one can compute in time polynomial in n' a coloured Muller game with n' nodes and m' colours such that, for all sufficiently large m, n,*

- $m' \leq (4m/\log(m)) \cdot \log(n)$,
- $n' \leq n^{(4m/log(m))+4}$ *and*
- *the given graph H has a dominating set of size up to m iff player Anke has a winning strategy in the resulting coloured Muller game.*

Proof. Assume that the given graph H has vertices $\{a_1, \ldots, a_n\}$ and let E be the set of its edges. Without loss of generality assume that $n, m \geq 2$ so that $\log(n), \log(m)$ are at least 1.

The main complexity bound in the parity game is due to some compression of m-tuples. Instead of giving a plain m-tuple of $m \cdot \lceil \log(n) \rceil$ bits, one stores the m-tuple in subsets of a base set E of colours.

Let $\{a_{k_1}, a_{k_2}, \ldots, a_{k_m}\}$ be Anke's choice of the dominating set in the graph H. The idea is to code each sequence of nodes $a_{k_1}, a_{k_2}, \ldots, a_{k_h}$ as a subset of $h \cdot \lceil \log(n)/\log(m) \rceil$ colours from a set E of $(2m+2) \cdot \lceil \log(n)/\log(m) \rceil$ colours. Note that for each subset E' of E with at most m colours, there is a one-one mapping $f_{E'}$ from $\{a_1, \ldots, a_n\}$ to subsets of E such that $f_{E'}(a_\ell)$ is the disjoint union of E' and a set of $\lceil \log(n)/\log(m) \rceil$ colours. This one-one function exists, as $|E - E'|$ has at least $\lceil \log(n)/\log(m) \rceil \cdot (m+2)$ elements and

$$|E - E'|!/(\lceil \log(n)/\log(m) \rceil! \cdot (|E - E'| - \lceil \log(n)/\log(m) \rceil)!)$$

$$\geq m^{\lceil \log(n)/\log(m) \rceil} \geq 2^{\log(m) \cdot \log(n)/\log(m)} = 2^{\log(n)} = n.$$

Next one inductively defines

- $f(a_{\ell_1})$ as $f_\emptyset(a_{\ell_1})$;
- for $h = 0, 1, \ldots, m-1$, $f(a_{\ell_1}, \ldots, a_{\ell_h}, a_{\ell_{h+1}}) = f_{E'}(a_{\ell_{h+1}})$ where $E' = f(a_{\ell_1}, \ldots, a_{\ell_h})$ and $\ell_1, \ell_2, \ldots \in \{1, 2, \ldots, n\}$.

Now the idea of the game constructed for the reduction is that Boris repeatedly asks Anke to 'find an edge in H from a given a_i into some a_j of the dominating set', for various values of a_i; this is represented in the game by Anke going through her dominating set $a_{k_1}, a_{k_2}, \ldots, a_{k_m}$ by inductively visiting nodes of the game corresponding to $a_{k_1}, a_{k_2}, \ldots, a_{k_h}$ for $h = 0, 1, \ldots$ until an h is reached with $a_{k_h} = a_j$. In case these answers

are inconsistent for some choices a_i and $a_{i'}$ that is, there is an h such that for both choices a_i and $a_{i'}$, Anke goes through nodes of the game corresponding to $a_{k_1}, a_{k_2}, \ldots, a_{k_{h-1}}$ but then goes through nodes corresponding to a_ℓ and $a_{\ell'}$, with $a_\ell \neq a_{\ell'}$, for a_i and $a_{i'}$ respectively, then Boris will ask Anke to do this repeatedly for an infinite number of times and win the game.

Now the constructed Muller game is defined more formally. For this, in addition to the colours E, the colours $\{d_0, d_1, \ldots, d_h\}$ are also used, where $E \cap \{d_0, d_1, \ldots, d_h\} = \emptyset$. Note that there are in total $(2m+2) \cdot \lceil \log(n)/\log(m) \rceil$ colours in E and $m+1$ colours in $D = \{d_0, d_1, \ldots, d_h\}$.

To simplify the presentation, in the description below a node w can have several colours from $D \cup E$, instead of only one colour. This convention does not affect the generality, as the node w can be replaced by a sequence of up to $|D \cup E| + 1$ nodes, where each node in the sequence has one colour from the set of colours assigned to w and only one choice for next move; one colour might need to be repeated in order to fix the player that moves next. All nodes will have at least the colour d_0, so that there are no colourless nodes.

Boris wins a play iff the colours of infinitely often visited nodes consist, for some h, exactly of $\{d_0, d_1, \ldots, d_h\}$ and at least $h \cdot \lceil \log(n)/\log(m) \rceil + 1$ of the colours in E. Note that Boris' winning condition from above is closed under union. Given two winning conditions U, U' for Boris, there are h, h' such that

- U has at least $h \cdot \lceil \log(n)/\log(m) \rceil + 1$ colours from E and the colours $\{d_0, d_1, \ldots, d_h\}$ and
- U' consists of at least least $h' \cdot \lceil \log(n)/\log(m) \rceil + 1$ colours from E and the colours $\{d_0, d_1, \ldots, d_{h'}\}$ and
- $U \cup U'$ has at least $\max\{h, h'\} \cdot \lceil \log(n)/\log(m) \rceil + 1$ colours from E and the colours $\{d_0, d_1, \ldots, d_{\max\{h,h'\}}\}$.

Thus $U \cup U'$ is also a valid winning condition of Boris. For this reason, it is sufficient to consider memoryless winning strategies of Anke. The nodes of the game and their colours are as follows:

1. Starting node $(start)$ with colour $\{d_0\}$, from where Boris moves to any node of the form $(edge, a_i)$ described in item 2.
2. For $i \in \{1, 2, \ldots, n\}$, node $(edge, a_i)$ with colour $\{d_0\}$. From node $(edge, a_i)$ Anke can move to nodes $(a_j, a_{k_1}, f(a_{k_1}))$ described in item 3 such that (a_i, a_j) is an edge in the graph H.
3. For $j, k_1, k_2, \ldots, k_h \in \{1, 2, \ldots, n\}$, node $(a_j, a_{k_h}, f(a_{k_1}, \ldots, a_{k_h}))$ with colours $f(a_{k_1}, \ldots, a_{k_h}) \cup \{d_0, d_1, \ldots, d_h\}$. Boris can move from node $(a_j, a_{k_h}, f(a_{k_1}, \ldots, a_{k_h}))$ to a node as described below:

(a) always to nodes $(edge, a_i)$ for i with $1 \leq i \leq n$;
(b) if $h < m$ and $a_j \neq a_{k_h}$, then also to the node $(nextround, a_j, f(a_{k_1}, \ldots, a_{k_h}))$, described in item 4;
(c) if $h = m \wedge a_j \neq a_{k_h}$, then also to the node $(fail)$ described in item 5.

4. For h with $1 \leq h < m$ and $j, k_1, k_2, \ldots, k_h \in \{1, 2, \ldots, n\}$, node $(nextround, a_j, f(a_{k_1}, \ldots, a_{k_h}))$ with colours $f(a_{k_1}, \ldots, a_{k_h}) \cup \{d_0, d_1, \ldots, d_h\}$. Anke can move from a node $(nextround, a_j, f(a_{k_1}, \ldots, a_{k_h}))$ to a node of the form $(a_j, a_{k_{h+1}}, f(a_{k_1}, \ldots, a_{k_h}, a_{k_{h+1}}))$, for $a_{k_{h+1}}$ being a node in H, as described in item 3. Here Anke should choose $a_{k_{h+1}}$ according to her dominating set in the graph H, as otherwise she would lose based on the winning conditions of Boris.
5. Node $(fail)$, with colours $D \cup E$, the players can only move from $(fail)$ to $(fail)$.

So in total there are $|E|+m+1$ colours and their number can be bounded by $m' = 4m \cdot \log(n)/\log(m)$ for all sufficiently large m. Furthermore, there are

- 1 node according to item 1,
- n nodes according to item 2,
- at most $2^{m'} \cdot n^2$ nodes according to item 3,
- $2^{m'} \cdot n$ nodes according to item 4 and
- 1 node according to item 5.

These $2^{m'} \cdot (n^2+n)+n+2$ nodes can be translated into at most $2^{m'} \cdot (n+1)^2 \cdot (m'+1)$ nodes in the regular Muller games, with one colour each, as indicated above. Using $m' \leq 4m \cdot \log(n)/\log(m)$ and $m \leq n$, this number of nodes can be bounded by $n^{4m/\log(m)} \cdot (n+1)^2 \cdot (4n\log(n)+1)$ which is bounded by $n' = n^{(4m/\log(m))+4}$ for all sufficiently large n. Thus, m' and n' are almost always upper bounds on the number of colours and nodes, respectively.

For the verification, it is shown that Anke wins the game iff there is a dominating set of size at most m in the original graph. If there is a dominating set $\{a_{k_1}, a_{k_2}, \ldots, a_{k_m}\}$, for each choice $(edge, a_i)$ made by Boris (whenever it is Boris's turn) Anke always chooses $a_j \in \{a_{k_1}, a_{k_2}, \ldots, a_{k_m}\}$ such that (a_i, a_j) is an edge in H and moves to $(a_j, a_{k_1}, f(a_{k_1}))$. Then, Anke goes through choices $(a_j, a_{k_h}, f(a_{k_1}, a_{k_2}, \ldots, a_{k_h}))$, for $h = 2, 3, \ldots$, until Boris chooses $(edge, a_{i'})$ in its turn. Note that this will happen when $a_{k_h} = a_j$ or before. The maximal h such that $a_j = a_{k_h}$ is chosen in the play, will make the set of infinitely often visited colours $U =$

$f(a_{k_1}, \ldots, a_{k_h}) \cup \{d_0, d_1, \ldots, d_h\}$, which has only $h \cdot \lceil \log(n)/\log(m) \rceil$ colours from E and therefore it will be a win for Anke and not for Boris.

In the case that there is no dominating set, the play will either end up in the node $(fail)$ or Anke would not be using its dominating set consistently as illustrated below. One assumes that Anke plays a memoryless winning strategy and that Boris knows Anke's strategy; thus it is sufficient to show that Boris can counteract this strategy. So assume that there is no dominating set of size m. There are two cases.

(a) Anke does not consistently maintain the same candidate for a dominating set. Then there are two nodes $a_i, a_{i'}$ such that, when Boris asks Anke to follow an edge from a_i and $a_{i'}$, respectively, into the dominating set then Anke's answers are not consistent. More precisely, there are a_i and $a_{i'}$, such that, in the case that the play goes to nodes $(edge, a_i)$ and $(edge, a_{i'})$, Anke (in series of moves via nodes of item 2 and 3) chooses $(*, a_{k_1}, f(a_{k_1}))$, $(*, a_{k_2}, f(a_{k_1}, a_{k_2}))$, $\ldots$, $(*, a_{k_{h-1}}, f(a_{k_1}, a_{k_2}, \ldots, a_{k_{h-1}}))$, and then chooses

$$(*, a_{k_h}, f(a_{k_1}, a_{k_2}, \ldots, a_{k_{h-1}}, a_{k_h})) \text{ or } (*, a'_{k_h}, f(a_{k_1}, a_{k_2}, \ldots, a_{k_{h-1}}, a'_{k_h})),$$

respectively, where $a_{k_h} \neq a'_{k_h}$ and $*$ can be any $a_j \in \{a_1, a_2, \ldots, a_n\}$. Boris exploits this inconsistency by moving to $(edge, a_i)$ at the start of the play and by alternately choosing $(edge, a_{i'})$ and $(edge, a_i)$ when the play enters

$$(*, a_{k_h}, f(a_{k_1}, a_{k_2}, \ldots, a_{k_h})) \text{ or } (*, a'_{k_h}, f(a_{k_1}, a_{k_2}, \ldots, a'_{k_h})).$$

As Anke plays a memoryless strategy, the play will result in an infinite loop of nodes with the colours

$$f(a_{k_1}, a_{k_2}, ..., a_{k_{h-1}}, a_{k_h}) \cup f(a_{k_1}, a_{k_2}, ..., a_{k_{h-1}}, a'_{k_h}) \cup \{d_0, d_1, ..., d_{h-1}, d_h\}$$

and the resulting play is won by Boris.

(b) Anke maintains consistently the same candidate for a dominating set $\{a_{k_1}, a_{k_2}, \ldots, a_{k_m}\}$. As this candidate is not a dominating set, there is some a_i such that no edge goes from a_i into this candidate set. So Boris chooses $(edge, a_i)$ and Anke moves to a node $(a_j, a_{k_1}, f(\{a_1\}))$. Now the play will go through all rounds and at each round through the node $(a_j, a_{k_h}, f(a_{k_1}, \ldots, a_{k_h}))$ and as $a_j \neq a_{k_h}$, Boris can continue the play up until it reaches $(fail)$. Then the play will remain in that node forever and as $E \cup D$ is a set of colours which is in the winning condition of Boris, Boris will win this play as well. $\square$

Note that the above algorithm reduces the question whether a graph of n nodes has a dominating set of size m to a Muller game with up to $m' =$

$(4m/\log(m))\log(n)$ colours and $n' = n^{(4m/\log(m))+4}$ nodes, for sufficiently large m and n. The construction of the game takes time $O(n')$. Assuming now that there is a decision procedure for Muller games which takes time $(2^{m'} \cdot n')^c$, for some constant c, one can obtain a decision procedure for the previous dominating set problem in time $n^{c\cdot((8m/\log(m))+4)}$, where the exponent is clearly $o(m)$. Chen, Huang, Kanj and Xia [10, Theorem 5.4] showed that if there is an algorithm which solves dominating set with these parameters – note that their paper uses a different notation – then $\mathbf{FPT} = \mathbf{W[1]}$. This provides the following corollary.

Corollary 14. *For every c there is no algorithm running in time $O((2^m \cdot n)^c)$ which decides the winner of a coloured Muller game with n nodes and m colours unless* $\mathbf{FPT} = \mathbf{W[1]}$.

The above reduction, due to the optimisation involved, increased massively the number of nodes in the game. If one does not want to lower the factor m to $(m/\log(m)) \cdot \log(n)$ in the translation, but only wants to achieve $O(m \cdot \log(n))$ colours, then there are more straightforward methods. Zielonka [46] used similar methods to show **NP**-hardness of the Muller games, even in the special case of games where player Anke, in the case that she wins, also has a memoryless winning strategy.

Theorem 15 (Zielonka [46]). *The problem to determine whether Anke can win a Muller game when the set of Boris' winning conditions is closed under union is* **NP***-complete; however, for containment in* **NP***, the winning conditions have to be represented as in Zielonka's paper. In general, this problem is in* $\mathbf{\Sigma_2^P}$.

Note that for games where Anke might win, but not with a memoryless winning-strategy, the complexity bound is worse. Dawar, Horn and Hunter showed that the problem to decide the winner in a Muller game is **PSPACE**-complete [11].

4 Memoryless Coloured Muller Games

This section is dealing with memoryless coloured Muller games. Memoryless games are games where Anke wins iff she (a) plays a memoryless strategy and (b) wins the game according to the specification of the game. If she does not do (a), this is counted as a loss for her. This was already done by Björklund, Sandberg and Vorobyov [3, Section 5] for Streett games and it can also be done for Muller games.

In contrast to normal coloured Muller games, the complexity of the

memoryless games is different. On one hand, one can decide in $\mathbf{\Sigma_2^P}$ whether Anke has a winning strategy: There is a memoryless strategy of Anke such that the graph obtained by fixing Anke's moves according to the strategy, does not allow Boris to reach a loop of length up to $2n^2$, where the set of colours of this loop is a non-member of G. On the other hand, the next result shows that unless **NP** can be solved in quasipolynomial time there is no analogue of the translation of Björklund, Sandberg and Vorobyov [3] from memoryless coloured Muller games into parity games.

Theorem 16. *Given a graph with n nodes and given a number $m \in \{1, 2, \ldots, n\}$, one can compute in time polynomial in n*

1. *a coloured Muller game $\mathcal{G}$ with $O(m \cdot n \cdot \log(n))$ nodes and $O(m \cdot \log(n))$ colours with the side-condition that Boris' winning condition is closed under union and superset and generated by a list of two-element sets of nodes and*
2. *a memoryless coloured Muller game $\mathcal{G}'$ with $O(m \cdot n \cdot \log(n) \cdot (\log(m) + \log\log(n)))$ nodes and $O(\log(m) + \log\log(n))$ colours, where the winning conditions for Boris can be given as an $Poly(m \cdot \log(n))$-sized explicit list,*

such that the graph has an m-clique iff Anke can win the game $\mathcal{G}$ iff Anke can win the game $\mathcal{G}'$.

Proof. Suppose (V, E) is the graph for the m-clique problem. Without loss of generality assume that $(a, a) \notin E$ for each $a \in V$ and that $(a, b) \in E \Leftrightarrow (b, a) \in E$ for all $a, b \in V$. Furthermore, assume that the nodes of V are represented using $\lceil \log(n+1) \rceil$ bits.

Construction of $\mathcal{G}$. The game $\mathcal{G}$ will use $1 + 2m \cdot \lceil \log(n+1) \rceil$ colours. The colours are 1 and $C^i_{b,k}$, for $i \in \{0, 1\}$, $b \in \{1, 2, \ldots, m\}$ and $k \in \{1, 2, \ldots, \lceil \log(n+1) \rceil\}$. Then, in part 2 this game will be modified to use only $O(\log(m) + \log\log(n))$ colours.

Intuitively, the game constructed below has m copies of the graph (V, E), with each copy having a subgraph (network) for each $a \in V$.

For each $a \in V$, $b \in \{1, 2, \ldots, m\}$, the nodes and edges of the game $\mathcal{G}$ are as follows:

(a) Nodes $(entry, a, b)$. These nodes are entry nodes for the subgraph related to node a in the b-th copy of (V, E). All the entry nodes have colour 1 and have Boris's turn to move.

(b) Nodes $(middle, a, b, k)$, for $k \in \{0, 1, 2, \ldots, \lceil \log(n+1) \rceil\}$. All these nodes are for Anke's move. There is an edge from $(entry, a, b)$ to nodes $(middle, a, b, k)$, for $k \in \{0, 1, 2, \ldots, \lceil \log(n+1) \rceil\}$. For $k = 0$, $(middle, a, b, k)$ has colour 1. For $k \in \{1, 2, \ldots, \lceil \log(n+1) \rceil\}$, the node $(middle, a, b, k)$ has colour $C^i_{b,k}$, where i is 1 if the k-th bit of a is 1 and 0 otherwise.

(c) Nodes $(transfer, a, b)$. All these nodes are with colour 1 and are for Boris's move. There is an edge from the node $(middle, a, b, k)$ to $(transfer, a, b)$ for $k \in \{0, 1, 2, \ldots, \lceil \log(n+1) \rceil\}$.

(d) Nodes $(last, a, b, b')$, where $b' \in \{1, 2, \ldots, m\}$. All these nodes have colour 1 and are for Anke's move. For each $b' \in \{1, 2, \ldots, m\}$, there is an edge from $(transfer, a, b)$ to $(last, a, b, b')$.

(e) There is an edge from $(last, a, b, b')$ to $(entry, a', b')$, for all a' such that there is an edge $(a, a') \in E$.

(f) There is a $(start)$ node in the game with colour 1 on which Anke moves. The edges from $(start)$ are to any node $(entry, a, 1)$, where $a \in V$. The game starts at $(start)$.

Note that $1 + 2m \cdot \lceil \log(n+1) \rceil$ colours are used in the game.

The winning condition for Boris is that, for some b, k, the game goes infinitely often through the colours $C^0_{b,k}$ and $C^1_{b,k}$. Note that this winning condition is closed under union and supersets and there is a polynomial time algorithm which checks whether a set U of colours evaluates to winner Anke or Boris.

Now, if there is a clique in the graph with nodes $a_1, a_2, \ldots, a_m$, then at (start), Anke can chose to go to $(entry, a_1, 1)$ and then from the nodes $(last, a_b, b, b')$, she can choose to go to $(entry, a_{b'}, b')$. For any fixed b, k, the nodes $(middle, a, b, k)$ that the game goes through have $a = a_b$. It immediately follows that only the colour $C^i_{b,k}$, with i being the k-th bit of a, can be visited infinitely often and not $C^{1-i}_{b,k}$. Thus, Anke wins.

On the other hand, if there is no clique, then Boris has the following winning strategy. Note that for any fixed sequence $a_1, a_2, \ldots, a_m$, of nodes, there are $s, t \in \{1, 2, \ldots, m\}$ such that (a_s, a_t) is not an edge in the graph (V, E).

Let $A_1, A_2, \ldots, A_r$ be all possible combinations of m nodes in V (r is roughly n^m).

Consider one round as going, for some a, b, b', k, through the following sequence of nodes: $(entry, a, b)$, $(middle, a, b, k)$, $(transfer, a, b)$, $(last, a, b, b')$.

In the rounds numbered $2p \cdot q$ and $2p \cdot q + 1$, for $p \in \{1, 2, \ldots, r\}$ and q is a natural number, Boris chooses b' during the visit to $(transfer, a, b)$

(where a, b are as in the corresponding round) as follows: For round $2p \cdot q$, he chooses $b' = s$ and for round $2p \cdot q + 1$, he chooses $b' = t$, where s, t are such that in the p-th combination of m nodes $A_p = a_1, a_2, \ldots, a_m$, (a_s, a_t) do not form an edge in (V, E). As Anke's strategy is memoryless, the choice made by Boris at the nodes of the form $(entry, a, b)$ does not effect what Anke chooses to play.

By the constraints on edges in the game as in (e) in the play above, there do not exist $a_1, a_2, \ldots, a_m$ such that almost always for any b and b', in $(last, a, b, b')$ Anke chooses to go to $(entry, a_{b'}, b')$. Hence, there exist b and a, a' with $a \neq a'$ such that the game goes through $(entry, a, b)$ and $(entry, a', b)$ infinitely often. But then, Boris can choose k such that a, a' are different in k-th bit, whenever the play goes through node $(entry, a, b)$ or $(entry, a', b)$; otherwise he can choose $k = 0$. Thus the game goes through colour $C^0_{b,k}$ and $C^1_{b,k}$ infinitely often and Boris wins.

Construction of $\mathcal{G}'$. The game $\mathcal{G}'$ is constructed by replacing in $\mathcal{G}$ the nodes $(middle, a, b, k)$ for $a \in V$, $b \in \{1, 2, \ldots, m\}$ and $k \in \{0, 1, 2, \ldots, \lceil \log(n+1) \rceil\}$, by the following nodes with corresponding colours:

(g) Nodes $(middleA, a, b, k, h)$, where

$$h \in \{1, 2, \ldots, \lceil \log(m+1) \rceil + \lceil \log(\lceil \log(n+1) \rceil + 1) \rceil\}.$$

All these nodes are for Anke's move. For $k = 0$, the colour of these nodes is 1; for $k > 0$, the colour of node $(middleA, a, b, k, h)$ is $4h + 2 + 2j + i$, where j is the h-th bit of b k (binary representations of b and k concatenated together) and i is the k-th bit of a. There is an edge from $(entry, a, b)$ to each node $(middleA, a, b, k, 1)$, for $k \in \{0, 1, 2, \ldots, \lceil \log(n+1) \rceil\}$.

(h) Nodes $(middleB, a, b, k, h)$, where

$$h \in \{1, 2, \ldots, \lceil \log(m+1) \rceil + \lceil \log(\lceil \log(n+1) \rceil + 1) \rceil - 1\}.$$

All these nodes are for Boris's move. Each node $(middleB, a, b, k, h)$ has the colour 1. Furthermore, for

$$h \in \{1, 2, \ldots, \lceil \log(m+1) \rceil + \lceil \log(\lceil \log(n+1) \rceil + 1) \rceil - 1\},$$

an edge goes from $(middleA, a, b, k, h)$ to $(middleB, a, b, k, h)$, and from $(middleB, a, b, k, h)$ to $(middleA, a, b, k, h+1)$.

(i) There is an edge from $(middleA, a, b, k, h)$ for $h = \lceil \log(m+1) \rceil + \lceil \log(\lceil \log(n+1) \rceil + 1) \rceil$, to $(transfer, a, b)$.

Thus, in effect instead of a play going through node $(middle, a, b, k)$, the play goes through the nodes $(middleA, a, b, k, 1)$, $(middleB, a, b, k, 1)$, $(middleA, a, b, k, 2)$, $(middleB, a, b, k, 2)$, ..., $(middleA, a, b, k, \lceil \log(m+1) \rceil + \lceil \log(\lceil \log(n+1) \rceil + 1 \rceil))$. The remaining nodes visited in a play stay the same as before. Now, the winning condition of Boris will be appropriately modified as follows: for some values of b, k, for each $i \in \{0, 1\}$, the play has to go infinitely often through *only* the colours 1 and $4h+2+2j+i$, for each $h \in \{1, 2, \ldots, \lceil \log(m+1) \rceil + \lceil \log(\lceil \log(n+1) \rceil + 1) \rceil\}$, where j is the h-th bit in the binary representation of $b\ k$. □

This reduction shows that if Theorem 6 would also hold for memoryless coloured Muller games, then one could show that the clique problem could be solved in polynomial time. For a sufficiently large k and for $p(n) = (n \cdot m \cdot \log(n) \cdot (\log(m) + \log\log(n)))^k$, via padding, one can increase the number of nodes to $p(n)$. As the number of colours is in $O(\log(m) + \log\log(n))$, this value is below $\log(p(n))$ for sufficiently large constant k. Now the assumed result would give $\mathbf{P} = \mathbf{NP}$; however, many people believe this is false.

Remark 17. Björklund, Sandberg and Vorobyov [3, Section 5] considered memoryless Streett games (called Quasi-Streett games in their paper) and showed that these are $\mathbf{W}[1]$-hard. This hardness transfers to memoryless coloured Muller games, the only modification needed for the proof is to colour the nodes such that two nodes have the same colour only if they appear for all Streett conditions (E, F) either both in E or none in E and either both in F or none in F. Thus, in the diagram [3, Figure 2] of that construction, one can use

(a) one colour for the nodes on Level 0, Level 2 and Level 5, as they do not show up in any condition,
(b) one colour for all nodes on Level 3, as they show up only jointly in one set of one Streett condition,
(c) one colour for all nodes on Level 4, as those show up jointly in only one set of one Streett condition,
(d) individual colours for the $2k + 1$ remaining nodes on other levels.

This gives in total $2k+4$ colours. Now one can translate the Streett condition into a Muller condition for this coloured Muller game in a straightforward way. Player Anke wins a play of that game iff she moves memoryless (a requirement on both sides) and the colours of the infinitely often visited nodes satisfy the translation of the Streett condition. Thus the original construction, which reduces the existence of independent sets of k vertices

to the existence of a winning strategy for Anke in the memoryless Streett games with $k+2$ conditions, carries over to a construction which reduces the existence of independent sets of k vertices to the existence of a winning strategy for Anke in the memoryless coloured Muller game with $2k+4$ colours. So the memoryless coloured Muller game is also $\mathbf{W}[1]$-hard.

Furthermore, the arguments in the remark and the result above also show that, if there would be a translation for the parameterised memoryless coloured Muller game into parity games with similar size and speed constraints as in the case of standard coloured Muller games, then one would get two unlikely outcomes: $\mathbf{W}[1] = \mathbf{FPT}$ and a quasipolynomial algorithm for the clique problem.

5 Conclusion

The progress reported in this paper shows that solving parity games is not as difficult as it was widely believed. Indeed, parity games can be solved in quasipolynomial time – the previous bounds were roughly $n^{O(\sqrt{n})}$ – and they are fixed parameter tractable with respect to the number m of values – the previously known algorithms were roughly $O(n^{m/3})$. These results are in agreement with earlier results stating that parity games can be solved in $\mathbf{UP} \cap \mathbf{co\text{-}UP}$ [30] and that there are subexponential algorithms to solve the problem [32]. In spite of the current progress, *the original question whether parity games can be decided in polynomial time still remains an important open question.*

The above results on parity games are then used to give an algorithm of runtime $O((m^m \cdot n)^5)$ for coloured Muller games with n nodes and m colours; this upper bound is almost optimal, since an algorithm with runtime $O((2^m \cdot n)^c)$, for some constant c, only exists in the case that $\mathbf{FPT} = \mathbf{W[1]}$, an assumption which is considered to be unlikely.

One might ask whether the results obtained for parity games permit further transfers to Muller games, for example, in the special cases where (a) player Anke can employ a memoryless winning strategy due to the special type of the game or (b) one does not permit player Anke to use other strategies than memoryless ones. Note that case (b) differs from case (a), as in case (b) the condition on using memoryless strategies can be restrictive while case (a) applies to Muller games of those types where one knows that "if Anke has a winning strategy then she has a memoryless winning strategy". Case (a) was analysed by McNaughton [36] and Zielonka [46]; it applies to Muller games where the winning condition of

player Boris is closed under union [46].

The above mentioned lower bound directly also applies to case (a). For case (b), the complexity class of the general problem is also in the polynomial hierarchy but not **PSPACE**-complete (unless **PSPACE** is contained in the polynomial hierarchy) as the decision problem for coloured Muller games; however, the algorithmic bounds are, even for relatively small values of m, expected to be worse, as one can code **NP**-hard problems into instances with logarithmically many colours; this is also reflected by the fact that coloured Muller games are in **FPT** while memoryless coloured Muller games are **W[1]**-hard.

Acknowledgments. The authors would like to thank Krishnendu Chatterjee, Sasha Rubin, Sven Schewe and Moshe Vardi for correspondence and comments.

References

1. Julien Bernet, David Janin and Igor Walukiewicz. Permissive strategies: from parity games to safety games. *RAIRO - Theoretical Informatics and Applications*, EDP Sciences, 36:251–275, 2002.
2. Dietmar Berwanger and Erich Grädel. Fixed-point logics and solitaire games. *Theory of Computing Systems*, 37(6):675–694, 2004.
3. Henrik Björklund, Sven Sandberg and Sergei Vorobyov. *On fixed-parameter complexity of infinite games.* Technical report 2003-038, Department of Information Technology, Uppsala University, Box 337, SE-751 05 Uppsala, Sweden.
4. Henrik Björklund, Sven Sandberg and Sergei Vorobyov. Memoryless determinacy of parity and mean payoff games: a simple proof. *Theoretical Computer Science*, 310(1–3):365–378, 2004.
5. Hans L. Bodlaender, Michael J. Dinneen and Bakhadyr Khoussainov. On game-theoretic models of networks. *Algorithms and Computation*, Twelfth International Symposium, ISAAC 2001, Christchurch, New Zealand, December 2001, Proceedings. *Springer LNCS*, 2223:550–561, 2001.
6. Hans L. Bodlaender, Michael J. Dinneen and Bakhadyr Khoussainov. Relaxed Update and Partition Network Games. *Fundamentica Informaticae*, 49(4):301–312, 2002.
7. Ashok K. Chandra, Dexter C. Kozen and Larry J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.
8. Krishnendu Chatterjee and Thomas A. Henzinger. Strategy Improvement and Randomized Subexponential Algorithms for Stochastic Parity Games. *Twentythird Annual Suymposium on Theoretical Aspects of Computer Science*, STACS 2006, Marseille, France, 23–25 February 2006, Proceedings. *Springer LNCS*, 3885:512–523, 2006.
9. Krishnendu Chatterjee, Marcin Jurdzinski and Thomas A. Henzinger. Quantitative stochastic parity games. SODA 2004, *Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 121–130, 2004.

10. Jianer Chen, Xiuzhen Huang, Iyad A. Kanj and Ge Xia. Strong computational lower bounds via parameterized complexity. *Journal of Computer and System Sciences*, 72(8):1346–1367, 2006.
11. Anuj Dawar, Florian Horn and Paul Hunter. *Complexity Bounds for Muller Games*. Manuscript, 2011.
12. Antonio Di Stasio, Aniello Murano, Giuseppe Perelli and Moshe Y. Vardi. Solving parity games using an automata-based algorithm. *Twentyfirst International Conference on Implementation and Application of Automata*, CIAA 2016, 19–22 July 2016, Seoul, South Korea, *Springer LNCS*, 9705:64–76, 2016.
13. Christoph Dittmann, Stephan Kreutzer and Alexandru I. Tomescu. Graph operations on parity games and polynomial-time algorithms. *Theoretical Computer Science*, 614: 97–108, 2016.
14. Rodney G. Downey and Michael R. Fellows. *Parameterised Complexity*. Springer, Heidelberg, 1999.
15. Olivier Finkel and Stevo Todorčević. The isomorphism relation between tree-automatic structures. *Central European Journal of Mathematics*, 8(2):299–313, 2010.
16. Olivier Finkel and Stevo Todorčević. A hierarchy of tree-automatic structures. *The Journal of Symbolic Logic*, 77(1):350–368, 2012.
17. Oliver Friedmann. An exponential lower bound for the parity game strategy improvement algorithm as we know it. *Logic in Computer Science*, LICS 2009, pp. 145–156, 2009.
18. E. Allen Emerson. Automata, tableaux, and temporal logics. *Proceedings of the Workshop on Logic of Programs*, *Springer LNCS*, 193:79–88, 1985.
19. E. Allen Emerson and Charanjit S. Jutla. Tree automata, μ-calculus and determinacy. *Annals of IEEE Symposium on Foundations of Computer Science*, pp. 368–377, 1991.
20. E. Allen Emerson, Charanjit S. Jutla, A. Prasad Sistla. On model checking for the μ-calculus and its fragments. *Theoretical Computer Science*, 258(1-2):491–522, 2001.
21. Oliver Friedmann and Martin Lange. Solving parity games in practice. *Automated Technology for Verification and Analysis*, Seventh International Symposium, ATVA 2009, Macao, China, 14–16 October 2009, *Springer LNCS*, 5799:182–196, 2009.
22. Jakub Gajarský, Michael Lampis, Kazuhisa Makino, Valia Mitsou and Sebastian Ordyniak. Parameterized algorithms for parity games. *Mathematical Foundations of Computer Science*, MFCS 2015. *Springer LNCS*, 9235:336–347, 2015.
23. Aniruddh Gandhi, Bakhadyr Khoussainov and Jiamou Liu. Efficient algorithms for games played on trees with back-edges. *Fundamentica Informaticae*, 111(4):391–412 (2011).
24. Erich Grädel. Model Checking Games; slide "A linear time algorithm for Game". *Spring School on Infinite Games and Their Applications.* 15–19 March 2005. `http://www.games.rwth-aachen.de/old/Events/Springschool/`.
25. Erich Grädel, Phokion G. Kolaitis, Leonid Libkin, Maarten Marx, Joel Spencer, Moshe Y. Vardi, Yde Venema and Scott Weinstein. *Finite Model Theory and Its Applications*. Springer, 2007.
26. Andrey Grinshpun, Pakawat Phalitnonkiat, Sasha Rubin and Andrei Tarfulea. Alternating traps in Muller and parity games. *Theoretical Computer Science*, 521:73–91, 2014.
27. Florian Horn. Explicit Muller games are **PTIME**. *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science*, FSTTCS 2008, pp. 235–245, *Dagstuhl Technical Reports*, 1756, 2008.

28. Paul William Hunter. *Complexity and Infinite Games on Finite Graphs*. PhD Thesis, University of Cambridge, Computer Laboratory Hughes Hall, 2007.
29. Hajime Ishihara, Bakhadyr Khoussainov. Complexity of some infinite games played on finite graphs. *Graph-Theoretic Concepts in Computer Science*, Twenty-Eighth International Workshop, WG2002, Cesky Krumlov, Czech Republic, 13–15 June 2002, Proceedings. *Springer LNCS* 2573:270–281, 2002.
30. Marcin Jurdzinski. Deciding the winner in parity games is in **UP** ∩ **co-UP**. *Information Processing Letters*, 68(3):119–124, 1998.
31. Marcin Jurdzinski. Small progress measures for solving parity games. *STACS 2000, Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science*, *Springer LNCS*, 1770:290–301, 2000.
32. Marcin Jurdzinski, Mike Paterson and Uri Zwick. A deterministic subexponential algorithm for solving parity games. *SIAM Journal on Computing*, 38(4):1519–1532, 2008.
33. Imran Khaliq and Gulshad Imran. Reachability games revisited. *Second International Conference on Advances and Trends in Software Engineering*, SOFTENG 2016, 21–25 February 2016, Lisbon, Portugal, Proceedings, International Academy, Research and Industry Association (IARIA), Wellington, DE 19810, USA, pp. 129–133, 2016.
34. Bakhadyr Khoussainov and Anil Nerode. *Automata Theory and its Applications.* Birkhäuser, 2001.
35. Dietrich Kuske, Jiamou Liu and Markus Lohrey. The isomorphism problem for omega-automatic trees. Proceedings of *Computer Science Logic*, CSL 2010, *Springer LNCS*, 6247:396–410, 2010.
36. Robert McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993.
37. Andrzej Wlodzimierz Mostowski. Games with forbidden positions. Technical Report 78, Uniwersytet Gdanski, Instytut Matematyki, 1991.
38. Jan Obdrzalek. Algorithmic analysis of parity games. PhD thesis, University of Edinburgh, 2006.
39. Viktor Petersson and Sergei G. Vorobyov. A randomized subexponential algorithm for parity games. *Nordic Journal of Computing*, 8:324–345, 2001.
40. Sven Schewe. Solving parity games in big steps. *FCTTCS 2007: Foundations of Software Technology and Theoretical Computer Science*, *Springer LNCS*, 4855:449–460, 2007; *Journal of Computer and System Sciences*, available online, 2016.
41. Sven Schewe. From parity and payoff games to linear programming. *Mathematical Foundations of Computer Science 2009*, Thirty-Fourth International Symposium, MFCS 2009, Novy Smokovec, High Tatras, Slovakia, August 24-28, 2009. Proceedings. *Springer LNCS*, 5734:675–686, 2009.
42. Frank Stephan. *Methods and Theory of Automata and Languages*. Lecture Notes, School of Computing, National University of Singapore, 2016. `http://www.comp.nus.edu.sg/~fstephan/fullautomatatheory.ps`.
43. Colin Stirling. Bisimulation, modal logic and model checking games. *Logic Journal of IGPL*, 7(1):103–124, 1999.
44. Wolfgang Thomas. On the Synthesis of Strategies in Infinite Games. *Twelfth International Symposium on Theoretical Aspects of Computer Science*, STACS 1995, *Springer LNCS*, 900:1–13, 1995.
45. Thomas Wilkie. Alternating tree automata, parity games and modal μ-calculus. *Bulletin of the Belgian Mathematical Society*, 8(2):359–391, 2001.

46. Wieslaw Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theoretical Computer Science*, 200:135–183, 1998.