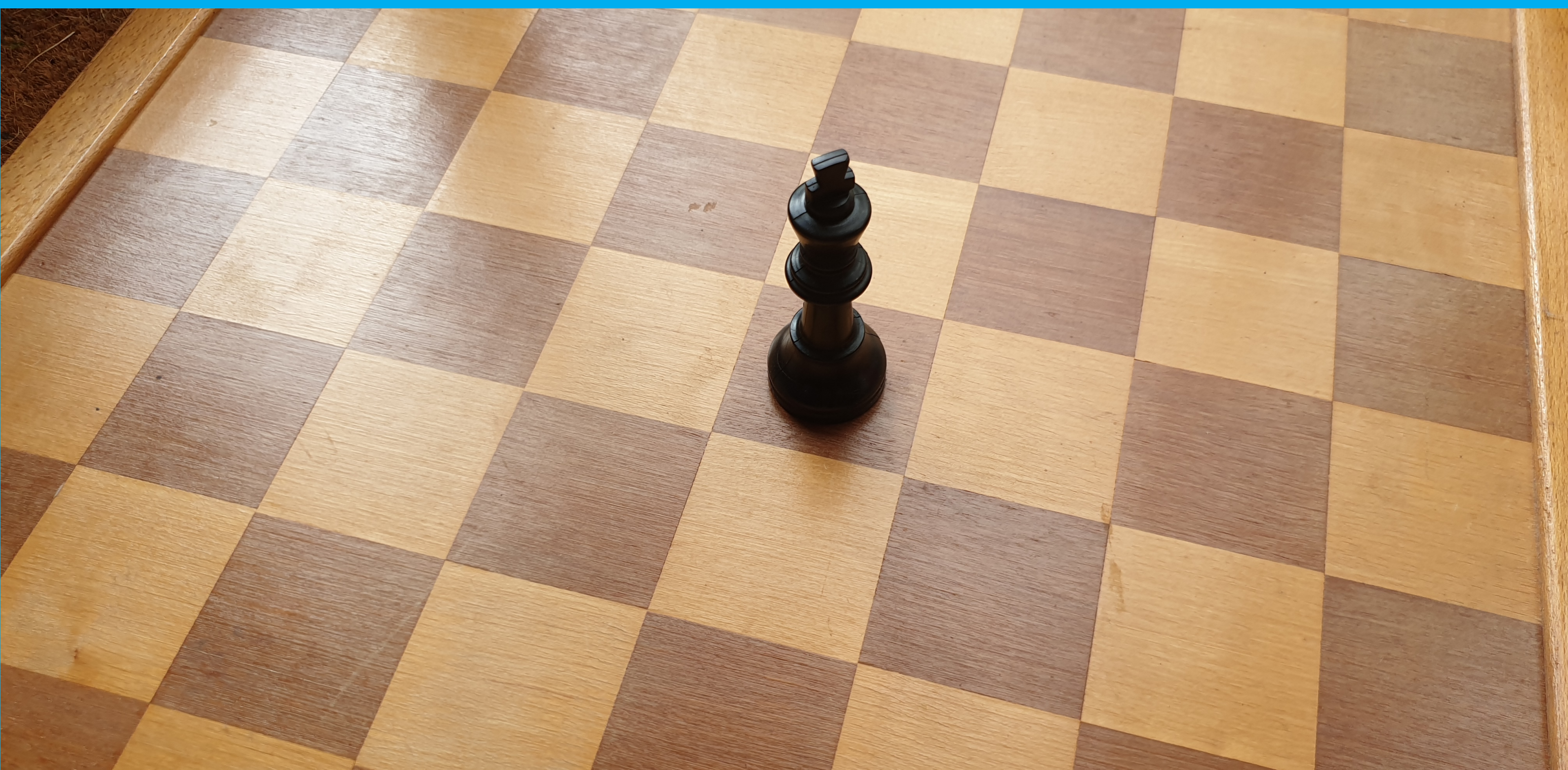
Domination Games on King Graphs

J. Op de Beek



TUDelft

Domination Games on King Graphs

by

J. Op de Beek

to obtain the degree of Bachelor of Science
at the Delft University of Technology,

Student number:	5387337	
Project duration:	April 22, 2024 – August 14, 2024	
Thesis committee:	Dr. C. Groenland,	TU Delft, supervisor
	Dr. C. Kraaikamp,	TU Delft, senior assessor
Special thanks to:	Dr. C. Groenland & P.P. Bastide	TU Delft
for their help during my bachelor end project.		

Abstract

The domination game is a two-player game played on a graph. The players have two roles, Dominator and Staller. Dominator wants to finish the game as soon as possible, while Staller tries to postpone the end of the game for as long as possible. The two players alternate turns and each turn they pick a vertex, and color the vertex itself and its neighbors. The players should always color at least one previously uncolored vertex. When all vertices are colored the game is finished. Calculating bounds on the total number of moves under optimal play for all graphs with n vertices is a hard open problem. In this research, the domination game on a specific family of graphs is analyzed, the king graphs. These are modeled after how kings move on a chessboard and provide an interesting graph with a lot of structure, but nontrivial optimal winning strategies. For small boards, we found the optimal move counts with computer search for up to 9×9 boards. For general n, lower and upper bounds on the number of moves were found. For an $n \times n$ king graph, the number of moves under optimal play, $\gamma_g(G_n)$, satisfies

$$\frac{2}{10}n^2 - O(n) \leq \gamma_g(G_n) \leq \frac{2}{9}n^2 + O(n).$$

J. Op de Beek
Delft, August 2024

Contents

1 Introduction

The *Domination Game* is a strategic two-player game in graph theory. In this game, two players, **Dominator** and **Staller**, take turns selecting vertices of a graph. The goal is to dominate all the vertices, meaning every vertex is either occupied by a player's piece or adjacent to an occupied vertex. The game ends when all vertices are dominated. The number of moves required under optimal play is called the *game domination number*, and when Dominator starts it is called $\gamma_g(G)$. When Staller starts, the game domination number is denoted by $\gamma'_g(G)$

- **Dominator**: Aims to dominate the graph using the fewest possible moves.
- **Staller**: Tries to delay domination, maximizing the total number of moves.

The game was invented and introduced in the paper by Brešar, Klavžar, and Rall [2] in 2010. They proved a few fundamental results, and stated some conjectures. The following conjecture came later and was published in Bujtás, Iršič and Klavžar [4] in 2020, and it is now a hot topic in the study of domination games.

1.1. The $\frac{1}{2}$ Conjecture

An important open problem in the study of domination games is the $\frac{1}{2}$ *Conjecture*. This conjecture states that for any graph G, which has minimum degree $\delta(G) \geq 2$, the number of moves when both players play optimally $\gamma_g(G)$ satisfies the inequality:

$$\gamma_g(G) \leq \left\lceil \frac{1}{2}|V| \right\rceil.$$

It makes sense to consider the number of vertices of the graph to make upper bounds, as generally the more vertices the more moves need to be played. The way the constant in this conjecture was chosen was by considering graphs C_n, simple cycles. They have degree exactly 2 everywhere, so they seem like the "sparsest" graphs, which should be hardest for Dominator, and a natural candidate for graphs that reach the extremal bound. And in fact the game domination number on simple cycles with $n \geq 4$ is exactly:

$$\gamma_g(C_n) = \left\lceil \frac{1}{2}n \right\rceil.$$

as can be seen in Watcharintorn Ruksasakchai and Worawannotai [8]

One recent advancement on this conjecture is by Portier and Versteegen [7] in 2023, which proves that:

$$\gamma_g(G) \leq \frac{10}{17}|V| + \frac{1}{17}.$$

This is close to $\frac{1}{2}$ but the constant is slightly larger. The methods to prove such bounds for different classes of graphs have not changed a lot, but the application of the methods became more complicated over time. One of the leading methods is the discharging method, which was already successfully applied to many problems in discrete mathematics before.

1.2. The Chessboard Variant

A particularly simple variation of the domination game is played on an $n \times n$ chessboard. This variation is a special case of the general game. Here, players place chess kings on the board, each king able to dominate all adjacent squares, including diagonals. The game itself remains the same. It is a nice special case to focus on, because the strategies for Dominator and Staller are quite rich, while the structure of the graph is very simple.

1.3. Algorithms and Complexity

Determining the minimum number of moves required for complete domination is a complex problem. The *Domination Game* is **NP-hard**, one of the fundamental results compiled in the book by Brešar, Henning, Klavžar and Rall [3], meaning it is computationally intensive to solve optimally for larger graphs. This means to find optimal solutions, even for small graphs that do not have extremely simple structure, the game domination number is easiest to calculate using techniques like exhaustive search, pruning, and heuristics to navigate the solution space effectively. A closely related topic to game domination numbers, is the domination numbers, which is basically the number of moves when the game is single player and only Dominator tries to end the game as soon as possible. *Exact algorithms* is the study of finding algorithms with usually exponential running time for NP-complete problems. This research will also try to find a fast exponential algorithm for finding the domination number, as one of the building blocks to build a fast computer search for finding the exact game domination number for small graphs.

1.4. Wider Picture

The study of domination games has important implications for both mathematics and computer science. By exploring these games, researchers gain insights into efficient algorithm development and develop new mathematical tools. Simple games such as the domination game, serve as a great playground where lots of ideas can be tested and progress can be measured quantitatively.

Understanding strategies and solutions for domination games also gives new tools to tackle different mathematical fields, with some advances in graph theory coming from some techniques that were developed for the domination game.

The study of game domination is very pure mathematics, and there have not been applications in other fields. But the way in which it provides a measure of progress, and it generates lots of interesting problems in discrete mathematics is enough reason to study it.

Overall, the domination game on graphs, and the chessboard variant for this research, serves as a valuable tool for exploring fundamental concepts in graph theory and algorithm design. Its study advances theoretical understanding, highlighting the importance of strategic thinking and computational efficiency in modern problem-solving.

1.5. Research Question

This research will completely focus on the domination game played on king graphs. The main research question is:

For an $n \times n$ king graph, what is the game domination number: $\gamma_g(G_n)$?

A few subquestions arise out of this main question:

- What is the exact value of the game domination number for small n?
- How to calculate those exact values of small n fast using a computer?
- What is a good upper bound on the game domination number?
- What is a good lower bound on the game domination number?

The next chapters will try to answer these questions. First of, chapter 2 quickly explains the exact workings of the domination game, assuming no prior knowledge. Then chapter 3 will cover the mathematical tools and definitions needed, which covers definitions from graph theory, and the mathematical definition of the domination game. Chapter 4 will show the mathematical and experimental results that try to answer the research question. And lastly chapter 5 will discuss the results, form new conjectures about the domination game on king graphs, and recommends further research.

2

Domination Games 101

This chapter serves as a gentle introduction to domination games, with no background knowledge required. Let's play a game. We start with an empty chessboard, and we will gradually place kings on the board, taking turns. I will start and place a king as shown in Figure 2.1. Note that the color of the king does not matter in this game.

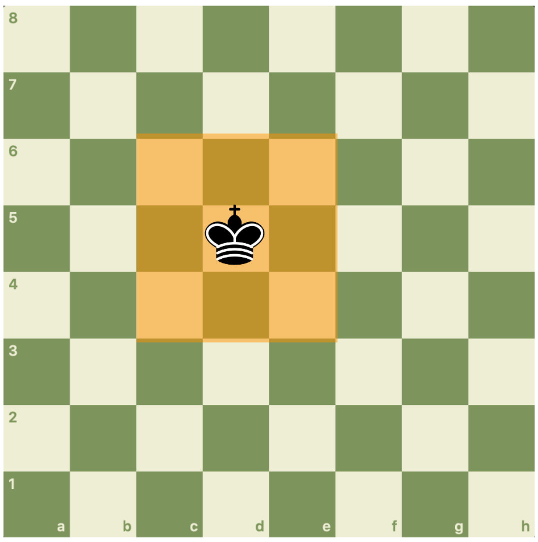


Figure 2.1: First move

I have colored the squares that the king can attack, and the square it is standing on in orange. These squares are now called dominated. My goal in the game is to dominate all of the squares as fast as possible, in the smallest total number of moves.

But now it's your turn, and you do the same thing, place a king somewhere on the chessboard, and colour the squares around it in orange, an example move can be seen in Figure 2.2

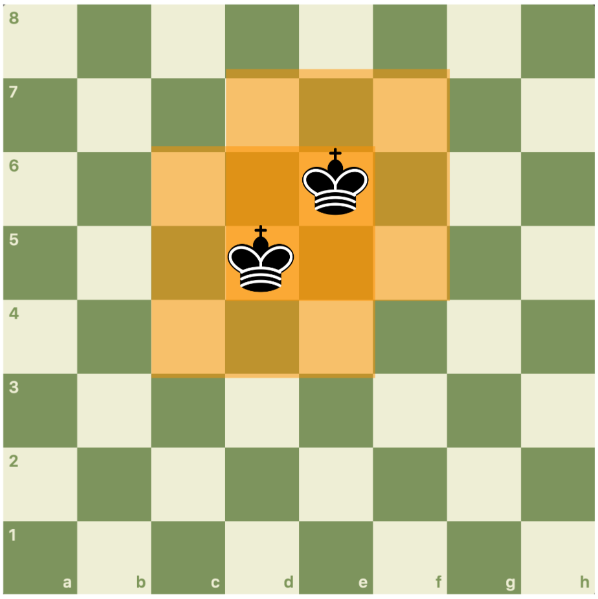


Figure 2.2: Second move

Your objective is to let the game last for as long as possible. You can try to achieve this, by making as few new squares orange as possible, and helping me the least. These two roles are called **Dominator** and **Staller** respectively, and they have opposite goals. There's one additional rule illustrated by Figure 2.3, which states that it is illegal to place a king in a position such that no new orange squares appear. This is to make sure Staller can't stall for a long time, by constantly placing kings in the middle of large orange regions. This would make the game boring. Note that kings that attack each other are allowed, as long as some fresh square is colored orange.

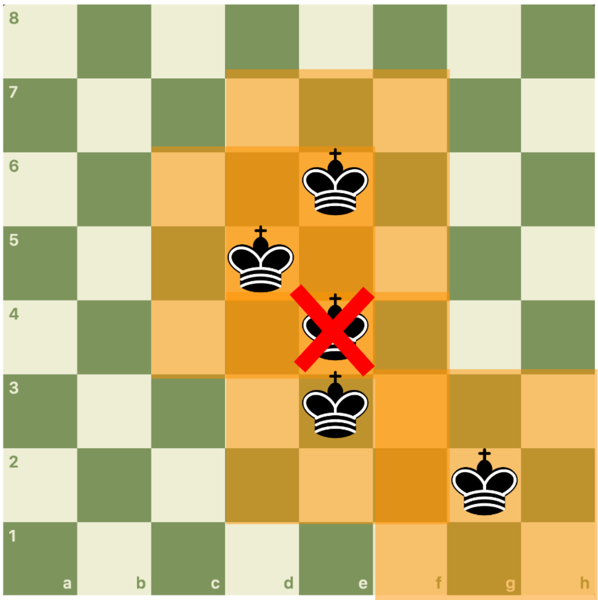


Figure 2.3: The king with a red cross, is an example of an illegal move. This move is illegal because no new squares are covered in this move.

Now we play on until there are no more uncolored squares, which can for example result in the board in Figure 2.4. By counting the number of kings, we can determine how many moves have been played, and what the final score is.

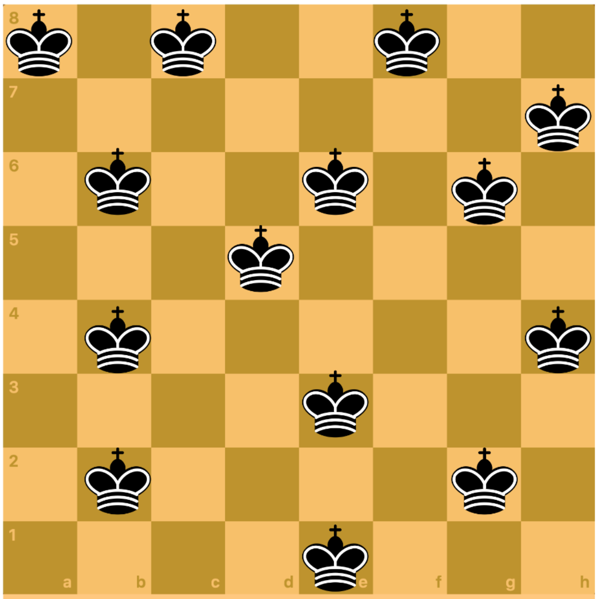


Figure 2.4: The game has finished now, because all the squares are orange. By counting the number of kings, we can see what the score is of the game, which I want to minimize, and you want to maximize.

The question is now, if we both try to play the best moves, how many kings end up on the board? This is the basic premise of domination games. While this is an example with a chessboard and kings, this same game can be played with different chess pieces, that cover different areas, such as knights or rooks. This would make the game a lot different, with maybe completely different strategies needed. In the branch of discrete mathematics, this game was central to quite some research papers. Usually the game is generalized to more complex boards, that do not have to be square, and in which the squares that are adjacent can also be varied. Still, mathematicians are interested in the basic question: how many moves will the game last when both players play optimally? But then the question is asked over all possible board shapes. We will stick to kings on chessboards.

2.1. Upper Bound

To get a feeling for how you can prove things about such a game, let's try to find an upper bound for the number of moves. This means,that no matter what Staller does, Dominator can always force the game to end in this many moves.

I will make a simple strategy that will say exactly which moves I will play. I will play my kings on the positions seen in Figure 2.5 , in some fixed order.

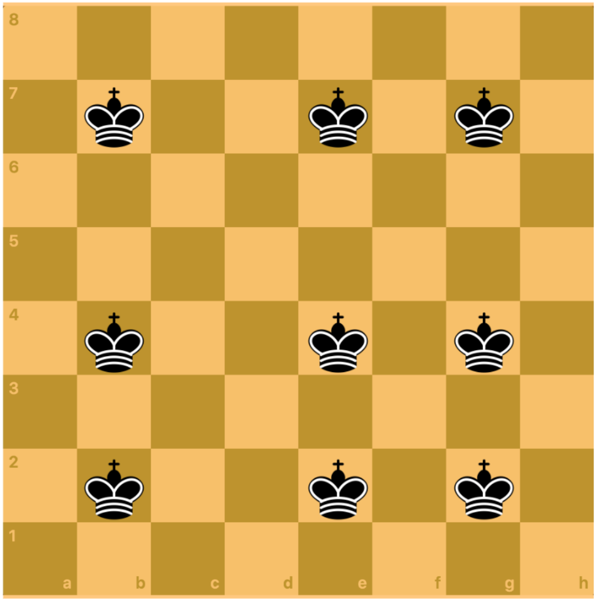


Figure 2.5: An example of a dominating set, a set of kings that dominates all the squares.

If you look closely, you can see that all squares of the chess board are attacked, so the game in this state is finished. Such an arrangement of kings is called a dominating set. Whatever Staller does, the game will finish at move 9 of Dominator, as Dominator has covered the board without any help from Staller.

The last missing piece of the puzzle, is that sometimes a move which Dominator wants to play will be illegal. This could potentially ruin the plan of Dominator. But instead of playing the illegal move, Dominator can skip this move, and go on with the next move in the order. The reason this works, is that the only way a move is illegal is if all the squares that it would color orange, are already colored. If this is the case, then Dominator does not need to worry about these squares, and does not have to play there. In total, with this strategy the number of moves played in total is at most $9+8=17$. These are 9 moves from Dominator, and 8 moves from Staller in between. This is a proof that the game domination number (this just refers to the optimal number of moves) is at most 17. The actual optimal number of moves happens to be 12, as seen in Table 4.1. So this bound is by no means the best upper bound you can give.

3

Preliminaries

This chapter will give an overview of all the tools and definitions that will be used in the rest of this report. These include the basic definitions of graphs and properties of graphs, the domination game and the associated domination game numbers and some results about how to bound domination game numbers.

3.1. Graph Definitions

- A simple, undirected graph $G = (V, E)$ is a set of vertices V, together with a set of edges E, connecting some pairs of vertices. Specifically the edges $\{u, v\} \in E$ are unordered pairs of two vertices from V. Note that selfloops (edges that connect a vertex with itself) are forbidden, and because E is a set, there cannot be multiple parallel edges. As a shorthand, the edge $\{u, v\}$ can be written as uv. Infinitely large graphs are disregarded in this paper, because playing domination games on them does not make much sense.

- The open neighborhood $N_G(u)$ of a vertex $u \in V$ in the graph G is the set of adjacent nodes $\{v \text{ s.t. } uv \in E\}$. When it is clear which graph is referred to, the subscript G can be omitted.

- The closed neighborhood is denoted by square brackets $N_G[u]$, for a vertex $u \in V$ and is almost the same as the open neighbourhood, but also contains the vertex u itself. So $N[u] = N(u) \cup \{u\}$.

- Both of these definitions can be extended to $N(S)$ and $N[S]$, for a subset of the vertices $S \subseteq V$. The are respectively the open and closed neighborhoods for a set of vertices. For open neighbourhoods, we take all the open neighbourhoods of the vertices, but do not take S. So it can be written as

  $$N(S) = \left(\bigcup_{u \in S} N(u) \right) \setminus S.$$

  While for closed neighbourhoods it is the more straightforward

  $$N[S] = \bigcup_{u \in S} N[u].$$

- A subset of the vertices $D \subseteq V$ is a dominating set, when $N[D] = V$.

- The domination number $\gamma(G)$ is the minimum size of any dominating set of graph G.

- The induced subgraph of a set of vertices $S \subseteq V$, $G[S]$ is a graph derived from the original graph G by taking

  $$G[S] = (S, \{uv \in E \text{ s.t. } u \in S \text{ and } v \in S\}).$$

3.2. Domination Game

Using the graph terminology in this chapter we can now define the domination game played on a graph G. Two players, Dominator and Staller take turns placing tokens on the graph. The goal of Dominator is to end the game in the fewest total number of moves, and Staller wants to extend the game for as long as possible. There are two different games defined where either Dominator or Staller begins.

A legal move consists of placing a token on a vertex of the graph. Let $D_{\text{old}} = N[S_{\text{tokens}}]$ be the set of dominated vertices before the move. Then D_{new}, the set of dominated vertices after the move, should be a proper superset of D_{old}, this rule applies to both Dominator and Staller . This rule makes sense, because otherwise Staller could always play a token on top of an already placed token, and the game would be boring, as it just devolves to Dominator playing the smallest dominating set.

We say that $\gamma_g(G)$ is the number of moves that get made in the game on graph G when both players play optimally, and Dominator starts. The subscript g is for differentiating between the game domination number and the domination number $\gamma(G)$. Similarly, $\gamma'_g(G)$ is the number of moves that get played in the game when Staller starts. These are called game domination numbers. Playing optimal means that a player tries to maximize the best outcome for himself, no matter what the responses of the other player would be. As this is a zero-sum game, this uniquely identifies the value of the game, the total number of moves that will be played in the end.

Sometimes we want to refer to the number of moves that still need to be made in a graph of which a subset $S \subseteq V$ of vertices is already dominated. In this case we can refer to $\gamma_g(G|S)$, which is the optimal number of moves in a game where you start with the vertices in S already dominated and Dominator starts. When Staller starts, $\gamma'_g(G|S)$ is again the number of moves under optimal play. Notice that in this definition, only the dominated vertices are given, and not the places where moves were played in the game. This is because the exact placement of moves does not matter for the state of the game after some moves, because the only rules for legal moves are about dominated vertices.

3.3. Bounds on Game Domination Number, through Domination Number

Given a graph G we want to find bounds on the game domination number $\gamma_g(G)$ in terms of the domination number $\gamma(G)$. Because domination numbers are generally much easier to calculate than game domination numbers, this gives an easy tool to get rough estimates. The following theorem by Brešar et al. [2] shows the connection between the two:

Theorem 3.3.1. *For any graph G:* $\gamma(G) \leq \gamma_g(G) \leq 2\gamma(G) - 1$.

Proof. For the first inequality we can notice that the vertices that are chosen during the domination game form a dominating set of graph G, otherwise the game would not be finished. Because $\gamma(G)$ is the minimum size of any dominating set, the domination game number must be at least the dominating number.

For the second inequality, let us fix any minimum size dominating set D. It will have $\gamma(G)$ vertices. We make a strategy for Dominator. In each move, Dominator chooses any vertex from the set D that is a legal move in the game. If such a vertex does not exist then all the vertices are already covered in the game. Why? Assume that some vertex is not yet dominated in the game. Then the set D contains some vertex that dominates this vertex, because D is a dominating set itself, and there is a legal move on one of the vertices of the dominating set, contradiction. Dominator cannot play more than $|D|$ moves this way, because afterwards the whole board is dominated, and he can never play the same vertex from D twice.

This means that Dominator plays at most $|D|$ moves, and Staller can play $|D| - 1$ moves in-between. In total, Dominator has a strategy that no matter what Staller does achieves $2|D| - 1 = 2\gamma(G) - 1$ moves. □

3.4. King Graphs

As a way to narrow this research to a manageable scope, and to make it more concrete, we will focus on playing domination games on king graphs. Given two integers $n > 0, m > 0$, the king graph consists of the vertices

$$V = \left\{(x, y) \in \mathbb{Z}^2 \text{ s.t. } 1 \leq x \leq n, 1 \leq y \leq m\right\},$$

which can be interpreted as the cells of a chessboard of a variable size. Two vertices of this graph are connected by an edge when a king on a chessboard can do a move from the first cell to the second cell. We refer

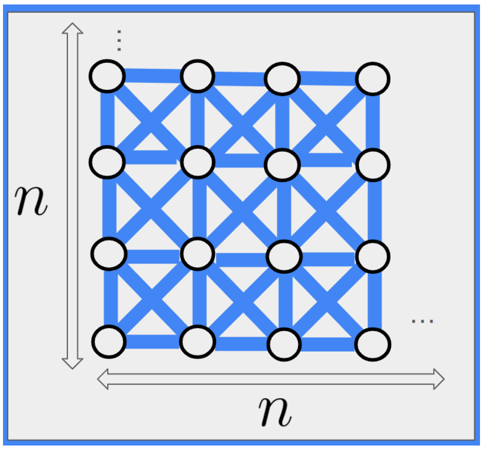


Figure 3.1: Illustration of an $n \times n$ king graph.

in the paper to the vertices of this graph as cells or squares. Formally, the edge set consists of:

$$E = \left\{\{(x,y),(x',y')\} \text{ s.t. } \max\left(|x-x'|,|y-y'|\right) = 1\right\}.$$

The domination game can now be interpreted as placing kings on an $n \times m$ chessboard, where the vertices that get colored, are the vertices that the king attacks. Note that this game has not much to do with chess, the color of the kings does not matter, and in principle kings can be placed next to each other, if this makes the dominated set larger. An illustration of a king graph can be seen in Figure 3.1.

3.5. Imagination Games

Imagination games already appeared in the seminal paper of Brešar et al. [2]. They form a cornerstone of proofs for the domination game, and are generally used to prove bounds on the number of moves in the domination game. Imagination games are used to generate a strategy for one of the players, while assuming the other player plays optimally. In this technique, one player plays the domination game on the original graph, and at the same time keeps in mind an imagined board, which could be slightly different from the original graph. When the player needs to choose a move to do in the original game, it checks what would be the best move in the imagined game, and tries to plays that move in the original game, with some extra rules for what should happen when this move would be illegal. When the opponent plays a move, it is played both in the imagined game and the actual game. If this would result in an illegal move in the imagination game, we choose some other move for the opponent. Any move the opponent does in the imagined game is only as good as the optimal move the opponent can play in the imagined game.

So if you can prove that an invariant holds which relates the dominated vertices of the imagined game and the actual game, such that one game always ends exactly at the same time or before the other, then the game domination number of the actual graph can be bounded with the game domination number in the imagined game.

3.5.1. Continuation Principle

The continuation principle is one of the simplest examples of applying imagination games to prove an important bound on the game domination numbers. The proof is omitted here but can be read in Brešar et al. [3].

Theorem 3.5.1. *Given a graph G and two subsets of vertices $S \subseteq V$ and $T \subseteq V$, such that $S \subset T$. Then $\gamma_g(G|T) \leq \gamma_g(G|S)$.*

This theorem essentially states that whenever the subset of dominated vertices is a superset of the dominated vertices of another state, the state of the game is more advantageous for Dominator. This seems intuitively clear, but it is not easy to prove from first principles. Using imagination games, it becomes a straightforward argument.

4

Results

4.1. Bounds on the Game Domination Number

In this section, we will see how to prove bounds on the game domination number on king graphs. Different proof strategies are tried to get the best bounds. The goal is to get bounds for large n, and see the big picture behavior of the game domination number $\gamma_g(G)$ for large king graphs.

4.1.1. Lower bound on game domination numbers of king graphs

A novel result we found in this research was a lower bound on the game domination number on king graphs. The proof for the lower bound gives a possible strategy for Staller. This strategy looks simple, and as seen in the Table 4.1, it often does not result in the best possible lower bound.

Theorem 4.1.1. *On a king graph of $n \times n$ cells, with $n \geq 22$, the game domination number satisfies:*

$$\gamma_g(G_n) \geq 2\left\lfloor \frac{1}{10}(n-1)^2 \right\rfloor.$$

Proof. We show a possible strategy for Staller to always achieve this number of moves. Let's define S_k, the set of dominated vertices after $k \geq 0$ moves have been played. An important quantity is the size of the set $|S_k|$ of dominated vertices. Clearly $|S_0| = 0$. Because the maximum degree in king graphs is 8, in each turn of Dominator, the size of the dominated set increases by at most 9, which is when a vertex and all its neighbours get dominated in one move. So for $a \geq 0$, $|S_{2a+1}| \leq |S_{2a}| + 9$. Now we want to make a greedy strategy for Staller, that dominates few new vertices at each of his turns. For this let's look at the lexicographically smallest undominated cell (x, y) (this is the cell with smallest x, and over those cells, the cell with smallest y coordinate, which is not yet dominated). There are 4 cases for the placement of this cell for the proof. For each case we give a candidate cell to place the next king of Staller.

- $(x, y) = (1, 1)$, Staller plays at cell $(1, 1)$
- $(x, y) = (1, y)$, for $y > 1$, Staller plays at cell $(1, y-1)$
- $(x, y) = (x, 1)$, for $x > 1$, Staller plays at cell $(x-1, 1)$ this is symmetric to the previous case.
- $x > 1$ and $y > 1$, Staller plays at cell $(x-1, y-1)$

Notice that all the moves Staller tries to play are legal moves, because they are inside the grid, and at least cover the previously uncovered cell (x, y). It's also clear that if none of the cases apply, the whole board is filled, because the cases cover the entire board and the game has ended. The first three cases could cover 4 or 2 new cells in Stallers move, which does not seem very advantageous, but the number of times these cases can occur can be bounded.

- $(1, 1)$ can only be covered one time.
- there are $n-1$ cells with $(1, y)$, for $y > 1$. But either $(1, 1)$ is covered by Dominater in which case $(1, 2)$ must also be covered, or in the first move of Staller $(1, 1)$ is taken. This means there are at maximum $n-2$ moves in which this case is played.

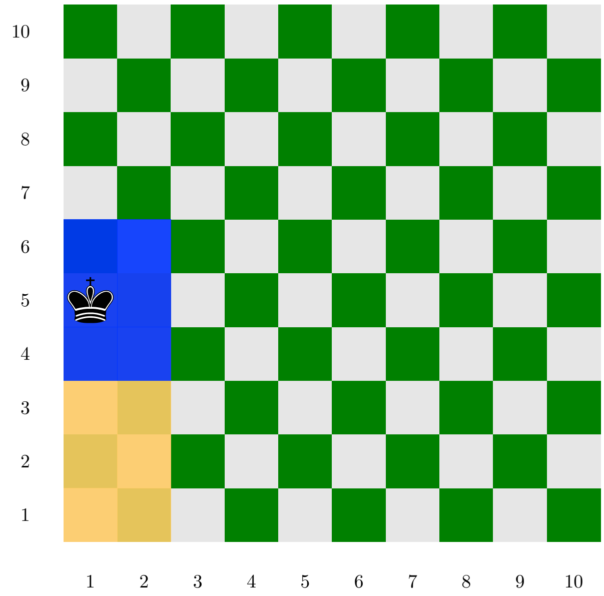


Figure 4.1: Illustration of Staller playing the lexicographically smallest square strategy. (1,6) is the first uncovered square, as the blue squares the king attacks cover up some orange squares. The strategy places a king on (1,5). It is clear that when a boundary square is orange, then the square one more to the centre, must also be orange, because a king can never cover a 1-wide area.

- With the same logic, the case $(x,1)$, for $x > 1$ happens at most $n-2$ times.

Together these cases happen at most $2n-3$ times, and the fourth case happens the rest of the time, as we made walls along $x = 1$ and $y = 1$ in the earlier moves. From here, we prove a bound on the maximum number of cells that can be covered at turn $2k$, with $k \geq 2n-4+1$. Now we look at how many new cells get covered in each of the cases

- When placing at $(1,1)$, at most 4 new cells get covered.
- When placing at $(x-1,1)$ or $(1,y-1)$ the situation is the same. So WLOG, just look at the case $(x-1,1)$. We know by the fact that we found the lexicographically smallest uncovered square, that $(x-1,1)$ and $(x-2,1)$ are covered. Due to the way king attacks are shaped, this also guarantees that $(x-1,2)$ and $(x-2,2)$ are covered. So at most 2 new cells can be covered. An example and a visual intuition for an example case can be seen in Figure 4.1
- When placing at $(x-1,y-1)$, it can be checked that all the squares that are covered by this move are lexicographically smaller than (x,y), which is the only newly covered square.

So in the worst case, we assume these cases all occur the maximum amount of times and add the most amount of new cells, and we get:

$$|S_{2k}| \leq 9+4+(9+2)\cdot(2n-4)+(9+1)\cdot(k-(2n-4+1)) \tag{4.1}$$

The game is over when $|S_{2k}| = n^2$. Now we can find a lower bound on the number of moves that the game will last, by finding the largest k in Equation 4.1 that makes this upper bound still smaller than n^2.

$$9+4+(9+2)\cdot(2n-4)+(9+1)\cdot(k-(2n-4+1)) \leq n^2 \tag{4.2}$$

Now this inequality can be solved for k, by placing all the constants and terms involving n on the RHS, and k on the LHS:

$$k \leq \frac{1}{10}(n-1)^2 \tag{4.3}$$

So $2\left\lfloor \frac{1}{10}(n-1)^2 \right\rfloor$ is the last integer, such that the upper bound on $|S_a|$ using Equation 4.1 is still less than or equal than n^2. So this proves that with the lexicographical smallest strategy the game will last for at least this many moves. We have to assume that $k \geq 2n-4+1$ for Equation 4.1 to be valid. This amounts to that the bound is only valid for $n \geq 22$. □

This n threshold for which the bound is valid, is quite large, and this is because although it is derived by checking when a quadratic function overtakes a linear function, the constants involved are quite large.

Note that for small $n \leq 21$ we can still use the same lexicographical smallest strategy. And the same ideas give a lower bound. If we also split into cases for odd and even number of moves, the bound can be made tighter. These bounds were used during the computer searches for speeding up the search and can be found in Section 4.4

This proof shows that most of the time Staller can only add 1 dominated cell for each of his moves. This is the minimum Staller can ever achieve on his moves, because a legal move covers at least 1 additional cell. But the strategy in the proof does not try to actively harm Dominator, and prevent him from always getting 9 new vertices. This is why I believe this is not the best lower bound for the game domination number.

4.1.2. General upper bounds on $\gamma_g(G)$

Theorem 4.1.2. *Given a disjoint partition of the nodes of a graph $G = (V, E)$ into k sets $S_1, S_2, \ldots, S_k$, define $T_i := N(S_i)$. T_i is the open neighborhood of S_i. Let's call the induced subgraph on the nodes $S_i \cup T_i$, or in other words $N[S_i]$, the graph G_i. Then we can bound the domination game number when Dominator starts as*

$$\gamma_g(G) \leq k + \sum_{i=1}^{k} \gamma'_{gm}(G_i).$$

$\gamma'_{gm}(G_i)$ denotes the domination game number of graph G_i when Staller starts, but a modified version of the game is played in which the T_i nodes are dominated from the start of the game.

Proof. We will show a strategy for Dominator that no matter what Staller does, achieves the bound of $k + \sum_{i=1}^{k} \gamma'_{gm}(G_i)$ moves.

We will use the concept of imagination strategies a lot. This means that Dominator will think of imaginary games on different graphs while playing the actual game, and make his decisions in the actual game, based on the state of the imagined graphs.

Dominater imagines k games, played on induced subgraphs

$$G_i := G[S_i \cup T_i]$$

In each of these games, it starts out with all nodes in T_i being already dominated. So players can still play moves on these nodes, but do not have to dominate them.

Superset Invariant

The invariant that holds after each move in the game on G is that the set of dominated vertices $D \subset V$ forms a (not necessarily proper) superset of the dominated vertices in the imagined games, that are not in T_i. In notation this becomes: $D \supseteq \left(\bigcup_{i=1}^{k} S_i \cap D_i\right)$. When all the D_i have become equal to S_i, it means the game in the original graph also must have finished, as the S_i form a partition of V and D is a superset.

Strategy

The strategy of Dominator will first give the initiative to Staller, and plays an arbitrary valid move, not moving anything in the imaginary games, this clearly preservers the superset invariant.

In each move of Staller, he will newly dominate at least one node, so let u be any node Staller newly dominates. Find the corresponding S_i, such that $u \in S_i$ and in the imaginary game on G_i play the same move as Staller played in the actual game. This is possible, because the graph G_i contains all neighbors of some node in S_i (this neighbor is either also in S_i, or it is in the open neighborhood T_i, and because D is a superset of $S_i \cap D_i$, u was also not dominated in G_i. This also preserves the superset invariant after the move, as only the nodes that are actually in S_i could get newly dominated, while in the actual graph, all these nodes will get dominated and maybe extra nodes outside S_i.

After each turn of Staller, there are two options for Dominator:

- Either Stallers move in the imagined game on G_i was the last move in this game.
- Or the game on G_i has not finished.

In the first case, Dominator again plays an arbitrary valid move. If at any point a valid move for Dominator is not possible, the game has now finished. It means the game only lasted shorter, so the bound on the number of moves still holds.

In the second case, Dominator looks at the last imaginary game G_i that Staller played in, and responds in this game with an optimal move. He tries to make the same move on the original graph. If this move is invalid, it means the set of vertices Dominator wanted to dominate was already dominated, so he again plays an arbitrary move.

For the same reasons as when Staller plays, playing on the same node in an imaginary game, as in the actual game preserves the superset invariant.

When all the imaginary games are finished, the original game is also finished. To bound the total number of moves, we note that in each imaginary game, Dominator always played optimal moves, and Staller starts, so the number of moves played in the game on G_i satisfies:

$$\text{moves played in the imagined game} \leq \gamma'_{gm}(G_i)$$

The only moves that Dominator plays that are not in correspondence with a move from one of the imagined games, are the starting move, and all moves where a game on G_i just ended. This adds potentially $1+k$ extra moves to the count, but after the last game G_i is finished, the game on G is finished, so Dominator does not have to respond the last time a game G_i is finished. This gives a total bound on the number of moves of

$$1 + (k-1) + \sum_{i=1}^{k} \gamma'_{gm}(G_i)$$

As this is a possible strategy for Dominator, it means the the number of moves under optimal play can only be less, which completes the proof. □

4.1.3. Applying the general bound to king graphs

To apply the bound from Theorem 4.1.2 to king graphs, we should choose the sets of the partition S_i in a smart way. Intuitively, having the sets T_i be small is great for getting bounds, as it means that in the imagined games in the proof, Staller has less opportunities to dominate few vertices, making the game last longer.

A natural candidate for S_i is to partition the king graph of $n \times n$, into subgrids of size $\ell \times \ell$, with ℓ small. The modified game on the graphs G_i is essentially the same as a king graph. This means that given some bounds for $\gamma'_{gm}(G_i)$, where G_i are $\ell \times \ell$ king graphs with some already dominated vertices at the edges of the grid (which corresponds to the set T_i), we can obtain a bound on king graphs for arbitrarily large n.

This is a nice idea in theory, but the theorem itself wastes one move for each subgrid, so when k is too small the bound obtained is not good. For large ℓ it's again hard to prove any good bounds. So the computational results obtained with a computer for small ℓ were not enough to improve over the bounds in the next section.

4.2. Domination number on king graphs

The general bounds in Section 3.3 show that the domination number can be a way to get upper and lower bounds for the game domination number. For king graphs, we can apply this bound if we know the domination number.

Lemma 4.2.1. *The domination number $\gamma(G_n)$ of an $n \times n$ king graph is $\lceil \frac{n}{3} \rceil^2$.*

Proof. First we show this number is a lower bound for the domination number: For a given king graph of size $n \times n$, take the set $S = \{(x, y) s.t. x = 1 \mod 3, y = 1 \mod 3) \text{ and } 1 \leq x, y \leq n\}$. In king distance, the minimum distance between any two distinct cells in set S is ≥ 3, which means that no two cells can be covered by the same vertex in a dominating set D. Thus $|D| \geq |S| = \lceil \frac{n}{3} \rceil^2$. The optimal D that achieves this lower bound can be made from S by moving every vertex in S from (x, y) to $(\min(n, x+1), \min(n, y+1))$. The set of cells that gets covered by any cell is basically a 3×3 square, and with this placement of kings, every cell is covered, visually this is pretty clear: Figure 4.2, and to fully verify it all the residues $\mod 3$ for x and y coordinates can be inspected, and boundary cases can be treated separately, but this is omitted here. This lower bound and construction prove the desired equality. □

So this immediately gives rise to some bounds on the game domination number on king graphs.

Corollary 4.2.2. *Combining 3.3 and 4.2.1, we get that for king graphs of side n: $\lceil \frac{n}{3} \rceil^2 \leq \gamma_g(G_n) \leq 2\lceil \frac{n}{3} \rceil^2 - 1$.*

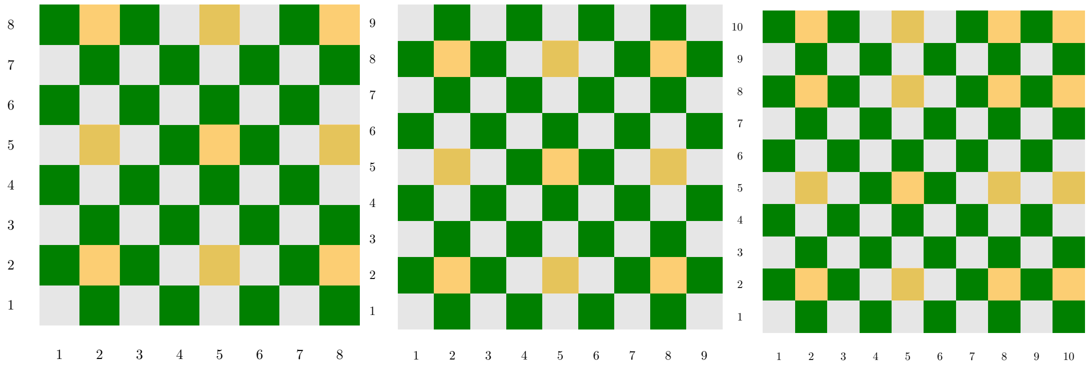


Figure 4.2: Dominating sets for all the cases modulo 3, boards are shown for $n = 8, 9, 10$.

4.3. Algorithm for Partially Dominated King Graphs

In the domination game, an important concept is that of the set S of already dominated vertices. And the game domination number $\gamma_g(G|S)$, given that this set S is already dominated. In such a state of the game, it may be useful to calculate the domination number $\gamma(G_n|S)$ which is the domination number given that we do not have to dominate the set S. Knowing this, we can use the exact same proof strategy as in Section 3.3 to find upper and lower bounds for the game domination number:

$$\gamma(G_n|S) \le \gamma_g(G_n|S) \le 2\gamma(G_n|S) - 1.$$

Calculating domination numbers in many types of graphs is NP-complete. This means for the king graph, with a set S of vertices that do not have to be covered, we cannot expect to use some general method that runs fast. Instead we can use Dynamic Programming, with a superpolynomial running time. This algorithm, together with a bound on the complexity of it, are found in Theorem 4.3.1.

Theorem 4.3.1. *The partial domination number of an $n \times n$ king graph, given a set of vertices T, $\gamma(G_n|T)$, can be calculated in $O\left(3^n poly(n)\right)$.*

Proof. First, we present the pseudocode of the algorithm in Algorithm 1.

Algorithm 1 Minimum Dominating Set on an $n \times n$ king graph, given that nodes in subset T are already dominated.

```
function MINDOMINATINGSET(int n, set T)
    table ← empty hashtable
    V ← {(i, j) ∈ ℤ² | 1 ≤ i ≤ n and 1 ≤ j ≤ n}
    function MINDOMINATINGSETREC(set S)
        if S ∈ table then
            return table[S]
        end if
        if S = V then
            return 0
        end if
        u ← least uncovered cell in lexicographic order
        ans ← ∞
        for each vertex v ∈ N[u] do
            ans ← min(ans, 1 + MINDOMINATINGSETREC(S ∪ N[v]))
        end for
        table[S] ← ans
        return ans
    end function
    return MINDOMINATINGSETREC(T)
end function
```

The algorithm is recursive, and recurses on subproblems of the form: Given that set S is already dominated, how many extra vertices need to be used to dominate the rest of the graph.

If all the cells are covered, we can safely return 0, and this is the base case of the algorithm.

Otherwise it finds the least uncovered cell in the $n \times n$ grid, we call it u. This cell has to get covered somehow, and all the cells in its closed neighborhood can be tried to cover this cell, there are at most 9 of them. In the smallest dominating set, one of these cells must dominate cell u, so taking the minimum over all these options, and recursively calling the function clearly gives the correct answer. The hashtable provides a way to stop the algorithm from calculating the subproblem of the same set S multiple times, by early returning from the function once the S is already found in the hashtable.

The time complexity of this approach is not obvious. We can prove a bound by bounding the number of subproblems S that will be searched, as in each subproblem only polynomial work is done. Let's split the dominated vertices S in the current subproblem into $S = S_{\text{prefix}} \cup S_{\text{fixed}} \cup S_{\text{active}}$.

- The set S_{prefix} denotes all the vertices that are already covered and are lexicographically smaller than the least uncovered cell. This set can only obtain $n^2 + 1$ different values, which are all the prefixes of the board, (the empty prefix, and the state in which $S = V$ also count).

- The set $S_{\text{fixed}} = T \setminus S_{\text{prefix}}$, where T is the original set of vertices for which we want to calculate the partial domination number. This set is uniquely determined when S_{prefix} is chosen, so this also does not contribute to any exponential running time.

- The set $S_{\text{active}} = S \setminus (S_{\text{fixed}} \cup S_{\text{prefix}})$ signifies all the cells that were dominated during the recursive algorithm, but are not yet part of the prefix of covered cells.

The active cells S_{active} must hug the boundary of the prefix set, and in each row they extend either 0, 1 or 2 steps from the S_{prefix} cells to the right. The reason for this is the way in which new kings are added when going over the 9 cases when going to a subproblem. A king always hits the lexicographically smallest uncovered cell, so it cannot extend more than 2 columns to the right from this cell. And there can also be no gaps, because of the square shape of the newly dominated area.

An example of this behavior of the active cells hugging the boundary of the prefix cells for $n = 10$ can be seen in Figure 4.3. This observation leads to a 3^n bound on the number of sets S_{active} given that the other two sets are fixed.

Therefore, there are at most $(n^2 + 1) \cdot 3^n$ subproblems that are visited, and at each subproblem, only polynomial time is needed. Applying dynamic programming, using an appropriate datastructure for storing all the intermediate results, such as a hashmap, the algorithm is proven to run in time $O(3^n \text{poly}(n))$.

One last subtle point, is that technically the active cells do not hug the prefix cells, when there is a cell from the fixed set S_{fixed} in the way. But this does not lead to more than 3 different patterns per row for the active set, so the complexity is unchanged. □

This dynamic programming technique on grids is not completely new, and the general paradigm of using dynamic programming, and the fact that the information stored as state in the table, stays "close" to the prefix of cells already considered is sometimes called *Broken Profile Dynamic Programming*. This is a term used mainly in competitive programming, and is explained in for example Aliyev and Dutt [1].

4.4. Calculation of Game Domination Numbers for small n

To gain more insight and data to guide making hypotheses, computer search was used to find the game domination number on king graphs, for small n. Different methods were tried to speed up the search.

4.4.1. Minimax

The most direct approach to calculate the game domination numbers is to recursively calculate $\gamma_g(G|S)$ and $\gamma'_g(G|S)$, the game domination number given that set S is already dominated. Depending on the player, the game domination number is either a minimum or a maximum over all the possible moves of the current player.

$$\gamma_g(G|S) = \min_{M \in \text{legal moves}} \left(\gamma'_g(G|(S \cup M)) + 1\right)$$

where M is the set of newly dominated vertices corresponding to a legal move. And for the game numbers when Staller is starting:

$$\gamma'_g(G|S) = \max_{M \in \text{legal moves}} \left(\gamma_g(G|(S \cup M)) + 1\right)$$

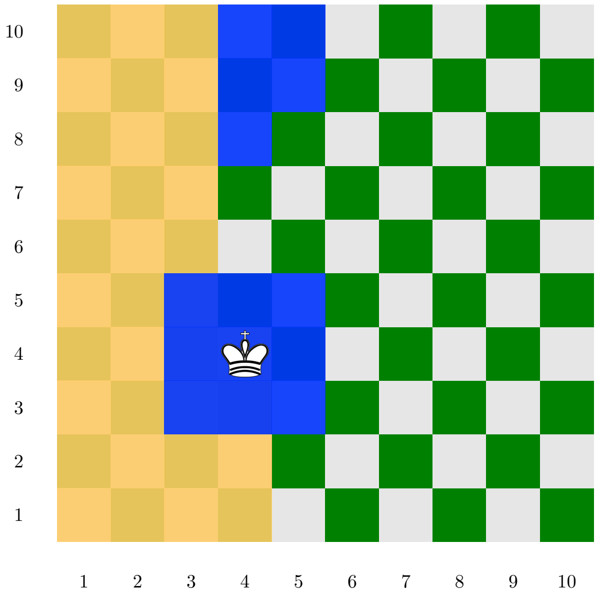


Figure 4.3: Running the partial domination algorithm. For reducing visual clutter, $T = S_{\text{fixed}} = \emptyset$ in this example. Here S_{prefix} is the orange region of cells, and some more cells that are now obstructed by the blue cells around the king. The cell $(4,4)$, before placing the white king was uncovered, and it was the least uncovered cell in lexicographic order. Before placing the white king, the blue cells at the top form the set S_{active}. The algorithm needs to choose an option of covering cell $(4,4)$ with a king, and one such option is shown in the figure. The blue cells around the king mark which cells it attacks. Because the least uncovered cell must be covered, the new blue squares must always connect to the yellow squares, and they cannot extend further than 2 squares to the right. So for each of the rows, the only option is for S_{active} to have 0,1 or 2 cells directly connected to the S_{prefix} orange cells.

When $S = V$, $\gamma_g(G|S) = \gamma'_g(G|S) = 0$

These formulas can be calculated recursively, and this will calculate the domination number in finite time, as each time the set S in recursive subproblems becomes bigger and the branching factor is bounded. The exploration of subproblems that minimax does, forms the game tree of the domination game. The name minimax comes from the fact that the recursive formulation alternately minimizes and maximizes, according to whose turn it is.

4.4.2. Basic memoization

To speed up the search, during the recursive evaluation of the different subproblems, we can maintain a hashtable with the answers that were already calculated for different subsets S, to not repeat the work of calculating these subproblems again. This is a simple form of Dynamic Programming.

When applied to king graphs of $n \times n$ cells, this naively gives an algorithm that works in $O(n^c 2^{n^2})$ time, for some constant c depending on the exact implementation, as there could be 2^{n^2} subsets being stored in the DP table. Luckily not all sets S are reachable from the starting state, so in actuality this algorithm can still solve $n = 7$ on a laptop in a few minutes. This is also the reason of using a hashtable as the memory usage of using an array of size 2^{n^2} would be prohibitive.

4.4.3. Optimizations

Without really changing the underlying search, the polynomial multiplicative factor of the algorithm can be improved. For the subset S of the cells that are dominated, we can use a bitset to optimize the space usage and runtime of iterating over moves and calculating the union of sets needed in the recursive formulas. A bitset is basically a way to store a bitstring (which has 1 if a cell is in the subset and 0 if a cell is outside the subset), in a concise way in the computer, by using the binary bits of an integer.

All the sets M that could be played by placing on a vertex, could also be precomputed, such that they do not have to be computed each time, and can also be stored in bitsets. Fast bitwise operations such as bitwise OR, also help to quickly find the union of two sets.

A simple way to cut some branches is by looking what the sets $S|M$ will look like when trying to calculate $\gamma_g(G|S)$ or $\gamma'_g(G|S)$. In the case of Dominator-first domination numbers, it does not make sense to play a move such that $(S \cup M_i) \subsetneq (S \cup M_j)$. According to the continuation principle, this will always lead to $\gamma'_g(G|(S \cup M_i)) \geq$

$\gamma'_g(G|(S \cup M_j))$. Symmetrically, the moves of Staller can also be pruned with the reversed $\subset$ relation. This makes the number of reachable states in the Dynamic Programming empirically a lot better.

4.4.4. Alpha-Beta Pruning

Another idea to decrease the number of states searched is to keep lower and upper bounds on what number of moves the two players can enforce. Let us define α as a lower bound such that Staller can always make the game last for at least that many moves. And β is an upper bound such that Dominator can always finish the game in at most this number of moves at the current state or one of the ancestors in the game tree. This method is not new, and for a more thorough exposition of Alpha-Beta pruning, see Knuth and Moore [6].

The basic way that this method improves over minimax, is that when α and β are equal during the recursive evaluation of subproblems, the recursion can immediately stop at this level, potentially cutting off a large chunk of the search space, without extra computation. To combine memoization and alpha-beta pruning, the α and β should be stored together with the game state in the hashtable, because the way in which to reach a subproblem with a subset S dictates what the α and β are.

4.4.5. Bounding α and β

Without even exploring the subtree of a subproblem we can already give some bounds on α and β. In particular the lower and upper bounds on the domination number on king graphs can be used, particularly the tighter version of Theorem 4.1.1 and Theorem 4.2.2. Given a subset S of dominated vertices, we need to find the domination number for the upper bound and we need to calculate the number of uncovered cells of the different cases described in the proof of Theorem 4.1.1 for the lower bound. The uncovered cells can be done with a simple linear scan, while calculating the domination number of a partially dominated king graph can be done with Theorem 4.3.1.

4.5. Results of computer search

These optimizations, and heuristics were added gradually during this research, and the largest n for which the exact game domination number could be found increased from 6 to 9. Eventually, these numbers could be found within 20 minutes on a laptop. For $n = 10$, the program was run on a desktop for some time, without result. But the main difficulty when $n = 10$ is that the memory usage becomes prohibitive. Maybe by using the Fat Nodes of the Delft Blue Supercomputer, this could be circumvented. Another way is to use a smarter caching strategy for the states that need to be cached, and the ones that are not. And a final improvement could be to parallelize the workload of this algorithm, to speed up the search by the amount of cores used, which seems doable as a lot of the work is embarrassingly parallel, although updating the α and β values correctly is not trivial. So this could be tried in further research. The final results can be found in Table 4.1

n	LB	$\gamma_g(n)$	UB	Iterations
1	1	1	1	1
2	1	1	1	1
3	1	1	1	1
4	3	5	7	47
5	5	7	7	235
6	7	7	7	1
7	9	12	17	62326
8	11	16	17	4466783
9	15	17	17	14739975

Table 4.1: With all the optimizations and heuristics listed in this chapter, the computer search could find the values of the game domination number for $1 \leq n \leq 9$. The algorithms runtime seems to grow rapidly, and based on the behavior for smaller n, $n = 10$ would be a tough challenge with just the techniques used here. The table shows the game domination number for different $n \times n$ king graphs. The LB and UB refer to the lower bound and upper bound for the game domination number, obtained by mathematical techniques earlier in this chapter. If these bounds are tighter it becomes easier for computer search to find the game domination number. An extreme case is $n = 6$, when the two bounds happen to give the same value. The last column gives the number of different boards the search algorithm has to search before concluding it has found the domination number. The 1 iteration that is needed for $n = 6$ is just the starting state, and it can immediately break because the lower and upper bound are the same.

5
Discussion

In this bachelor project, the domination game on king graphs was studied through the lens of both mathematics and computer science.

Despite efforts, getting lower and upper bounds for the domination game number that match each other remains a hard task. Currently the bounds proved for an $n \times n$ king graph are roughly:

$$\frac{2}{10}n^2 - O(n) \leq \gamma_g(G_n) \leq \frac{2}{9}n^2 + O(n).$$

Here, $\gamma_g(G_n)$ denotes the game domination number of a king graph.

There is still a significant gap between the lower and upper bounds which grows proportional to $\Theta(n^2)$. The two bounds follow from a general bound given the domination number, and a simple looking strategy for Staller. But both seem hard to improve, although they are straightforward.

Different ideas from the ideas presented in the results were tried, but did not seem to lead anywhere. Some method of proofs were looking at the connected components of black cells, or dividing the board in sections in a smart way to bound the number of moves per section. Another avenue of research was trying to find the domination numbers for King Graphs on a $3 \times n$ board, but the results for small n computed by the program were erratic.

Some ideas which did lead somewhere, but which required some care did not make it into the results section, because of time constraints. They are interesting enough to be included, but because the result is not complete, it is now given as an conjecture.

5.1. Partial Domination Numbers of King Graphs

5.1.1. NP-hardness

Given that Theorem 4.3.1 presents an exponential algorithm for calculating the domination number of a partially dominated king graph, one might question whether exponential time is the best possible solution. If $P \neq NP$, achieving a polynomial-time algorithm is likely impossible. This assertion is based on the idea of reducing an NP-complete problem on planar graphs to the problem of partial domination on king graphs. Although the main ideas of the proof are clear, the details require more care to achieve full rigor. Therefore, I propose the following conjecture:

Conjecture 5.1.1. *The decision problem of deciding whether for a king graph of size $n \times n$ and a subset of already dominated vertices S, whether the domination number $\gamma(G_n|S) \leq k$ is NP-complete.*

Proof Idea:

We reduce the Vertex Cover problem on planar graphs of degree 3, proven to be NP-complete by Das and Goodrich [5], to the partial domination problem on king graphs. The key idea is to transform a planar graph into a sufficiently large king graph, where vertices are represented by "smart gadgets" of 3×3 cells. These gadgets are designed such that they can be covered under specific conditions.

In the transformed king graph, most cells are already dominated, leaving only the gadgets with undominated cells. The gadgets can take the form of a U-shape (3×3) that can be covered in two ways: by placing a king in the middle, requiring one king, or by multiple kings covering parts of the U from each of the three sides.

By creating paths of spaced-out uncovered cells, we can establish connections between the gadgets using these paths, effectively simulating the degree 3 graph. When all three neighbors of a U-shape gadget are covered, it allows us to skip placing a king on the gadget itself, as it becomes covered by all three sides. This mechanism forces the positions where a king is placed in the middle of the U to correspond to a vertex cover.

Since planar graphs can be embedded in the plane using only vertical and horizontal lines, this construction serves as an approximate proof.

5.1.2. Better complexity estimates

Another way to improve the current results on the partial game domination number of king graphs algorithm, is to more carefully analyze if the actual number of active sets S_{active} is as high as the theorem Theorem 4.3.1 says. Because of the convex shape of a 3×3 square inside a grid, I think the exponential factor should be lower than 3^n, and closer to 2.5^n, because not all of the combinations of active cells on each row are reachable.

5.2. New Conjecture

Based on the results of the computer search an interesting conjecture can be made. For n that are multiples of 3, $n \in \{3,6,9\}$, each time, the game domination number is equal to the simple upper bound as seen in Table 4.1. This upper bound is solely derived from the domination number. It seems that in such boards whatever Dominator does, Staller can just avoid to make any progress on the domination of the board.

Conjecture 5.2.1. *For $n = 3k$, with $k \in \mathbb{N}$, the game domination number on an $n \times n$ king graph is equal to* $\frac{2}{9}n^2 - 1$

If the heuristics and search can be improved to also find the answer for $n = 12$, maybe this conjecture can be disproved. But together with the fact that progress on the upper bound mathematically was challenging, I believe it is a good guess.

If this conjecture would be true, the domination number up to $O(n)$ terms would also be known for all $n \times n$ king graphs. The upper bound provided by the domination number, would tight up to $O(n)$. And Staller can play the game on a smaller grid which is the closest n s.t. $n \mod 3 = 0$, using an imagination game, to also show that the game domination number is lower bounded by $\frac{2}{9}n^2 - O(n)$ for all n.

So what this conjecture says is that the lower bound on the game domination number on king graphs can be greatly improved, while the upper bound is tight.

5.3. Further Research

A possible way to book more progress on king graphs is to try to use more sophisticated methods such as the discharging method used in Portier and Versteegen [7] modified to work better for king graphs. The charging method assigns values to cells, based on their local surroundings, and the proof verifies that the sum of the charges, drops quickly, which means that Dominator can finish the game quickly. But trying to use such methods does mean hoping that Conjecture 5.2.1 is not true, because charging methods are usually trying to find better strategies for Dominator and not for Staller.

If somehow much better methods are developed for computer search and finding game domination numbers for larger and larger, but constant n, the partition bound in Theorem 4.1.2 can be used to get bounds for all n.

5.4. Place in Literature

There have been several papers about special cases of the Domination Number. Some of these get the exact game domination number for their special class of graphs such as Watcharintorn Ruksasakchai and Worawannotai [8]. They give an inductive proof of the game domination numbers, and the class of cycles and paths are just simple enough, that with enough effort this can be done. Analyzing king graphs, it seemed harder to come up with exact bounds, due to the 2D nature of the game, instead of 1D, for paths and cycles.

In terms of the open problems in the field of domination games, such as the $\frac{1}{2}$ conjecture, which states that the domination number never exceeds $\frac{1}{2}|V|$ for all graphs with the smallest degree at least 2, not much progress has been made. A part that is applicable from this research is the computer search code, and heuristics, which maybe could be used to look for counterexamples on some other classes of graphs. But the general consensus is that the $\frac{1}{2}$ conjecture is true, so this may not be fruitful.

Bibliography

[1] Silap Aliyev and Aditya Dutt. Dynamic programming on broken profile. Problem "Parquet" - Algorithms for competitive programming. URL https://cp-algorithms.com/dynamic_programming/profile-dynamics.html.

[2] Boštjan Brešar, Sandi Klavžar, and Douglas F. Rall. Domination game and an imagination strategy. *SIAM Journal on Discrete Mathematics*, 24(3):979–991, 2010. doi: 10.1137/100786800. URL https://doi.org/10.1137/100786800.

[3] Boštjan Brešar, Michael A. Henning, Sandi Klavžar, and Douglas F. Rall. *Domination Games Played on Graphs*. Springer International Publishing, 2021. doi: 10.1007/978-3-030-69087-8.

[4] Csilla Bujtás, Vesna Iršič, and Sandi Klavžar. 1/2-conjectures on the domination game and claw-free graphs, 2020. URL https://arxiv.org/abs/2010.14273.

[5] Gautam Das and Michael T. Goodrich. On the complexity of approximating and illuminating three-dimensional convex polyhedra. In Selim G. Akl, Frank Dehne, Jörg-Rüdiger Sack, and Nicola Santoro, editors, *Algorithms and Data Structures*, pages 74–85, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. ISBN 978-3-540-44747-4.

[6] Donald E. Knuth and Ronald W. Moore. An analysis of alpha-beta pruning. *Artificial Intelligence*, 6(4): 293–326, 1975. ISSN 0004-3702. doi: https://doi.org/10.1016/0004-3702(75)90019-3. URL https://www.sciencedirect.com/science/article/pii/0004370275900193.

[7] Julien Portier and Leo Versteegen. Progress towards the 1/2-conjecture for the domination game, 2023. URL https://arxiv.org/abs/2301.05202.

[8] Kritkhajohn Onphaeng Watcharintorn Ruksasakchai and Chalermpong Worawannotai. Game domination numbers of a disjoint union of paths and cycles. *Quaestiones Mathematicae*, 42(10):1357–1372, 2019. doi: 10.2989/16073606.2018.1521880. URL https://doi.org/10.2989/16073606.2018.1521880.