A Review of Dominating Set Problems

JESSICA PAQUETTE, CSC 422

1 INTRODUCTION

The *minimum dominating set problem* (MDS) is an NP-hard problem with multiple applications and several interesting properties. Despite its intractability, heuristics, approximation and exact algorithms have been developed for MDS on general graphs. We will also explore the applications of dominating set to chessboard problems, and open questions involving dominating set. Also discussed are several subsets of MDS including the *minimum connected dominating set*, *minimum cardinality dominating set*, and $r-$*dominating set* problems.

2 BACKGROUND

2.1 Graphs, Digraphs, and Multigraphs

A *graph* is a collection of *vertices* and *edges*. The set of vertices $V(G)$ for a graph $G = (V, E)$ consists of the labels for each vertex in the graph. The set of edges $E(G)$ for G consists of unordered vertex pairs $(u, v), u \neq v$ denoting that u is connected to v. Two vertices u and v in $V(G)$ are said to be *adjacent* if (u, v) is a member of $E(G)$. A vertex v is said to be *incident* to an edge e if v is one of the elements of e.

The *degree* of a vertex v is the number of edges incident to it. The notion of degree allows us to define the *maximum degree* of a graph, denoted Δ, as the highest degree of any vertex in the graph. A vertex is called *isolated* if its degree is 0. A graph is said to be *weighted* if its edges or vertices have a *weight* or *cost* associated with them, typically represented as a number.

A *digraph* (*directed graph*) is a collection of vertices and *arcs* (*directed edges*). Digraphs are distinguished from graphs in that arcs are ordered pairs rather than unordered. In domains where digraphs are common, "graphs" may be referred to as undirected graphs.

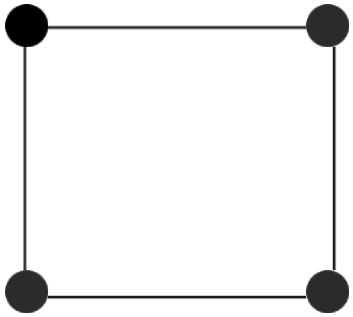

Figure 1: A *cycle graph* on four vertices.

A *multigraph* is a graph with a multiset of edges rather than a set. Multigraphs also allow for edges of the form (u, u) called *loops*.

2.2 Dominating Set Problems

A set of vertices D from a graph G is known as a *dominating set* if every vertex in $V(G)$ is either in D or is adjacent to some vertex in D. The *minimum dominating set of* G is the smallest dominating set for G.

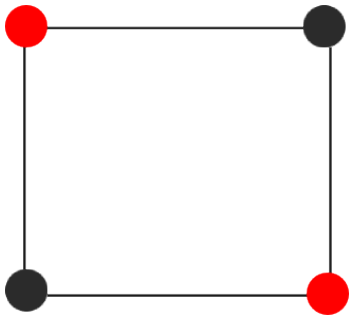

Figure 2: A minumum dominating set of figure 1 with its vertices coloured in red. Note that the black vertices also constitute a dominating set in this case.

The number of vertices in the minimum dominating set is called the *dominating number of* G, denoted $\gamma(G)$. Finding the dominating set of a graph refers to finding the minimum dominating set of that graph. The dominating set problem is *NP-hard*; there is no known polynomial-time

algorithm which finds the minimum dominating set for an arbitrary graph.

3 APPLICATIONS OF THE DOMINATING SET PROBLEM

Dominating set problems arise in several problem domains. Outlined are three problems which can be solved in natural or novel ways using dominating set.

3.1 Networks

3.1.1 Data Centre Allocation on a Network

Designing a computer network requires effective and efficient allocation of data centres to clients. A natural solution to this problem is to define a maximum distance ρ between two nodes and to minimize the number of centres on the network. This is called the *ρ-dominating set* problem. [1]

3.1.2 Wireless networks

The structure of wireless networks allows for a very natural representation as a graph. We let the vertices of the graph represent the communication devices on the network. For example, a radio transceiver would be a vertex on a radio network. Two vertices on a wireless network are adjacent if they are able to communicate with each other. [2]

Dominating sets on such networks can be used to model nodes that can broadcast information and have it reach the entire network. [2] The effective use of such a dominating set ensures that only broadcasting nodes have to forward a packet over the network, allowing non-broadcasting nodes to simply receive that given packet. [3] Dominating sets are also used for routing and data aggregation in computer networks. [3]

3.2 Error Correcting Codes

Given the set of Gaussian integers $\mathbb{Z}[i] = \{a + bi \mid a, b \in \mathbb{Z}\}$, it is possible to define a metric on the nodes of a circulant graph arising from $\mathbb{Z}[i]$. The circulant graphs have degree four and are known

as *Gaussian graphs*. Perfect error-correcting codes can then be generated by providing a solution to the *t-dominating set problem* for a collection of Gaussian graphs corresponding to constellations of Gaussian integers. [4]

Below is an image from [4] which shows a Gaussian graph and its isomorphic circulant.

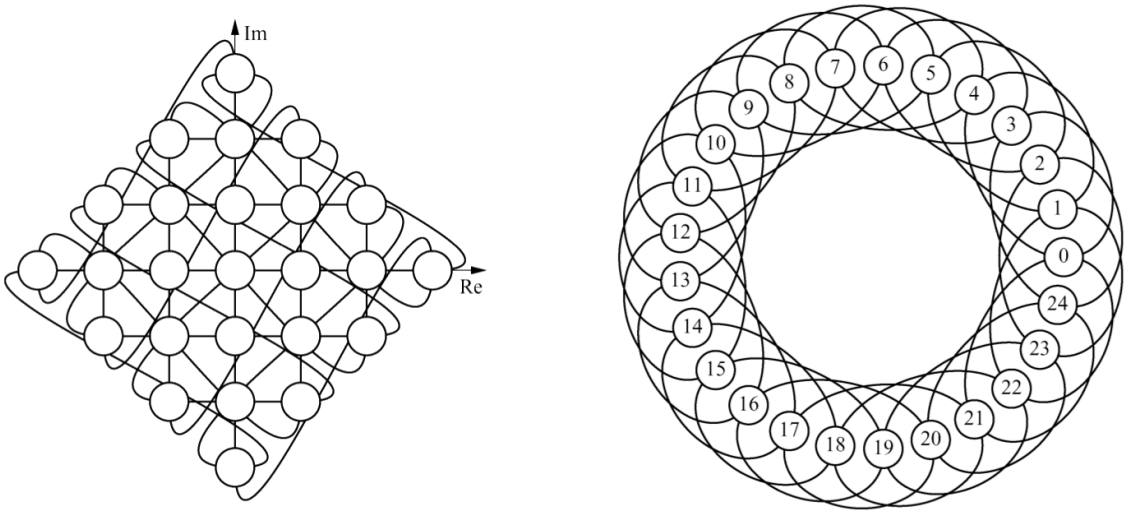

Figure 3: (a) The layout of G_{3+4i} (b) The circulant $C_{25(3,4)}$, which is isomorphic to G_{3+41} [4]

3.3 Social Networks of Individuals with Social Issues

A *positive influence dominating set* (*PID*) on a graph $G = (V, E)$ is a subset of $V(G)$ with the property that any vertex of G has at least $\lceil \frac{deg(v)}{2} \rceil$ neighbours in the PID. We can consider a person who requires an intervention to be an individual with a "social issue". The PID problem is shown to be APX-hard. [5]

A group of individuals with some given social issue, such as drug-addiction or binge-drinking, this can be modelled as a graph where the vertices are people, and the edges are friendships between people. Placing some of these individuals into an intervention program would make them a positive influence on their peers. If it is impossible to have every person in the group join the intervention program, then the minimal PID for the social network acts an an effective model to ensure that every member of the network is influenced by a positive force. [5]

4 EXACT DOMINATING SET ALGORITHMS

Although dominating set is a hard problem, there is value in being able to solve it with exact accuracy for some problem domains. The design goal of an exact exponential algorithm is to achieve better performance than an exponential search. [6]

4.1 Breaking the $\Omega(2^n)$ barrier for MDS on General Graphs

Fomin *et al.* present a search tree pruning algorithm that solves the minimum dominating set problem on arbitrary graphs in $O(1.93782^n)$ time. [7]

This is achievable through a restricted search space due to Reed [8]:

Theorem 4.1 *Every graph with $|V(G)| = n$ with minimum degree at least three has a dominating set of size at most $\frac{3n}{8}$. [8]*

The algorithm removes vertices of degree one and two as it branches into subcases. At the end of its execution, by Reed, the final graph G' has degree 3 or 0. Let V_3 refer to the vertices in $V(G')$ whose degree is at least 3, and let $t = |V_3|$. Due to Reed, there is a vertex set in the graph induced by V_3 on G' of size at most $\frac{3t}{8}$ dominating that graph. Given any subset X of $V(G)$, we can then say that this bound also holds for $X \cap V(G)$. Therefore, we only need to test the subsets with at most $\frac{3t}{8}$ vertices in order to find the minimum dominating set. Thus, we only have to test $\binom{t}{\frac{3t}{8}}$ sets. By Stirling's approximation for factorials, the running time of the algorithm is then $O(1.93782^t)$. [7]

4.2 Measure and Conquer

Recent efforts by F. Fomin *et al.* have resulted in a method called *measure and conquer* [9] which aids in providing a better analysis on Branch and Reduce-based exponential algorithms. The measure and conquer technique allows the programmer to use a nonstandard measure on a subproblem's size in order to bound the amount of time the program spends on each subproblem. They prove an $\Omega(2^{0.396n})$ lower bound on the general minimum dominating set problem.

The algorithm works by reducing the problem to an instance of minimum set cover with size at most $2n$. The set cover problem consists of a universal set $\mathcal{U}$ and a set of subsets of $\mathcal{U}$ called $\mathcal{S}$. The goal of the minimum set cover problem is to find the smallest set of members of $\mathcal{S}$ that contains every element of $\mathcal{U}$. They prove that using exponential space and measure and conquer that their branch and reduce-based minimum set-cover algorithm runs in $O(2^{0.305(|\mathcal{U}|+|\mathcal{S}|)})$. From this finding, they determine that the running time of their minimum dominating set algorithm is in $O^{0.598n}$

4.2.1 Measure and Conquer for Connected Dominating Set

The connected dominating set problem raises some interesting results on the complexity of a problem changing under certain restrictions. A *connected dominating set* (CDS) D on a graph $G = (V, E)$ is a dominating set under the restriction that the subgraph of G induced by D is connected. Connected dominating set, unlike the general dominating set, can be solved in better than $O(2^n)$ time; F. Fomin, F. Grandoni, and D. Kratsch present an $O(1.9407^n)$ algorithm for the connected dominating set problem using the measure and conquer technique. [10] Although a subset of the dominating set problem, the connected dominating set problem requires use of different algorithm design techniques than the general dominating set problem, as it is considered *non-local*.

Fomin, Grandoni, and Kratsch designed their algorithm with an invariant stating that at every step, the solution must remain connected. Their algorithm makes use of the Branch and Reduce paradigm; that is, it applies reduction rules at each step, and then recursively branches and works on the resulting subproblems. The measure and conquer technique is used to break the 2^n lower-bound. Also noted is an $\Omega(4^{\frac{n}{5}})$ lower-bound for their algorithm.

E. Camby and O. Schaudt note that given the connected dominating set problem, if $\gamma_c(G)$ is the *connected dominating number of* G, then $\frac{\gamma_c(G)}{\gamma(G)} < 3$. [11]

4.3 Parameterized Approaches

Parameterization is a common approach to coping with intractible problems such as dominating set. Parameterizing a problem refers to partitioning the problem into subsets according to the value of some input parameter. [12] Then the goal becomes to design an algorithm which is polynomial in the length of the parameterized inputs, but is not in the value of the parameterized input itself.

An example of a parameterization for minumum dominating set is to reduce it to a $k-$dominating set problem. Solving multiple instances of the $k-$dominating set problem would then allow us to solve the minimum dominating set for our graph. Unfortunately, this parameterization still yields a hard problem for general graphs. [13, 14]

4.3.1 RWB-Reduction Method

G. Philip *et al.*show that the $k-$dominating set problem is fixed-parameter tractable and has a polynomial-size kernel for graphs that do not have $K_{i,j}$ as a subgraph for $j \geq i \geq 1$. [13]

They construct a *rwb*-graph from the original graph G. The *rwb*-graph partitions $V(G)$ into three disjoint vertex sets R_G, B_G, and W_G of red, blue, and white vertices respectively. Then they parameterize by asking the answer to the *k-rwb-dominating set* problem on the constructed graph. [13]

They apply reduction rules at each step of the algorithm to decide how each vertex should be coloured, whether or not they should add *gadget vertices* at any step, and when to change the size of k. In order to yield the desired dominating set, the graph is then unperturbed to its original state through a colour-removal routine. The result is a kernel on $O((d+2)^{d+3} \cdot k^{(d+1)^2})$ vertices and edges which can be found in $O(2^d \cdot d \cdot n^2)$ for a d-degenerate graph with n vertices. [13]

4.3.2 Parameterization on Graphs of Bounded Genus

Ellis et. al extend the notion of *domination* to dominating some subset V' of the vertices in $V(G)$ for some graph G rather than requiring that all of $V(G)$ be dominated. First, they define $B \subset V(G)$ which contains some vertices that need to be dominated. They then create a flag *exists* which is true when there is a dominating set of at most size k for B. [14] The reduction steps used for the algorithm make sure that the graph is bounded by genus rather than by degree. A colouring method is used to define vertices as either *white* (those vertices not in B) or *black* (the vertices in B). The reduction procedure is found to be in $O(n^2)$. Additionally, any *white-black* reduction has a vertex of degree $4g + 39$ where g is the genus of the graph. By this result, after the reduction procedure, the algorithm runs in $O((4g+40)^k \cdot n^2)$

5 HEURISTICS AND APPROXIMATIONS

MDS is MIN $F^{+}\Pi_2$-complete, meaning that they can at best be approximated within some ratio of $log(n)$. However, certain restricted subsets of MDS yield far more accurate approximations. [15] Section 6.1 surveys a paper by M. Damian-Iordache and S. V. Pemmaraju which outlines a $(2+\epsilon)$-approximation for MDS on circle graphs.

5.1 Greedy heuristics

L. A. Sanchis constructs five greedy heuristics to compute a minimum dominating set and tests each one for effectiveness. Each one works by constructing a set D which is either "grown" or "shrunk" greedily into a dominating set. [16]

We will cover the ideas and results from each one separately.

5.1.1 Classical Greedy

The *classical greedy* heuristic starts with an empty dominating set D. At every step the algorithm finds the vertex v in $V(G)$ which would dominate the most previously undominated vertices and adds it to D. In the case that there is another vertex $u \neq v$ that dominates the same number of vertices as v, then the algorithm chooses between them at random. This heuristic runs in $O(nd+m)$ where d is the size of the dominating set. [16]

5.1.2 Reverse Greedy

The *reverse greedy* heuristic starts with $D = V(G)$. At every step, the algorithm picks a vertex $v \in D$ such that D will still dominate $V(G)$ after the removal of v. This implies that the vertex v cannot *uniquely* dominate any other vertex u. The vertices of the graph are initially sorted in ascending order. This way, we can quickly find the vertex with the lowest degree that fits the invariant. Like in the classic greedy heuristic, if two or more vertices fit the invariant, the algorithm will randomly pick one to remove from D. This heuristic runs in $O(nd+m+nlogn)$ where d is the size of the dominating set. [16]

5.1.3 Random Greedy

Starting with an empty D, each of the vertices in the graph is weighted by the number of vertices that it would dominate if it were chosen at the current step of the algorithm. A new vertex v is added to D with probability $\frac{weight_v}{\sum_{u \in N(v)} weight_u}$. The running time of this heuristic is comparable to the classical greedy heuristic. [16]

5.1.4 Greedy Vote

Like the random greedy heuristic, the *greedy vote* heuristic assigns weights to the vertices. However, greedy vote concerns itself with the *best* way to cover some vertex out of several possible ways. We note that every vertex v can cover at most $1 + deg(v)$ other vertices. We assign each vertex a *voting weight* of $\frac{1}{(1+deg(v))}$. Then we define the *total weight* of the v to be the sum of its own voting weight and the voting weights of its neighbours. Whenever the algorithm picks a vertex to be in D, it picks the one with the highest total weight. The complexity of this heuristic is the same as the classical greedy heuristic. [16]

5.1.5 Greedy Vote with Local Search

This heuristic uses the greedy vote routine and then performs an exhaustive local search to determine if it is possible to remove two vertices from D and replace them with one or zero vertices from $V(G)$. The local search is run on D until it is not possible to remove two vertices from D. The local search causes this heuristic to run in $O(n^2d^2)$. [16]

6 DOMINATING SET PROBLEMS ON SPECIAL CLASSES OF GRAPHS

This section explores approaches to solving dominating set problems within the domain of certain classes of graphs. Both approximation algorithms and exact algorithms are explored.

6.1 Circle Graphs

Let $G = (V, E)$ be a graph, and let C be a set of chords in a circle called an *chord intersection model*. Then G is a circle graph if there is a one-to-one mapping $\phi : V(G) \to C$, and ϕ ensures that two vertices u and v are adjacent if and only if their corresponding chords c_1 and c_2 intersect. [17, 18]

M. Damian-Iordache and S. V. Pemmaraju present a $(2 + \epsilon)$-approximation algorithm for MDS on circle graphs. [17] They later show that domination problems in general are APX-hard. [18]

6.1.1 Necessary Definitions

Before discussing the methods of Damian-Iordache and Pemmaraju, we need to introduce some specialized terminology for circle graphs.

Definition 6.1 *Given a circle graph G, the set of chords C is called a* ***chord intersection model*** *for G. [18]*

A second, equivalent definition of the circle graph is also necessary.

Definition 6.2 *A* ***circle graph*** *has a one-to-one mapping $\psi : V(G) \to I$ where I is a set of intervals where ψ ensures that two vertices u and v are adjacent if and only if their corresponding intervals i_1 and i_2 overlap, but neither contains the other. [18]*

Like how we defined the chord model before, we can now define an *interval model*.

Definition 6.3 *Given a circle graph G, the set of intervals I is called an* ***interval model*** *for G. [18]*

The main idea behind the algorithm is to solve a simpler problem known as the *leaf sector cover* problem, which can be computed in polynomial time. From this, it is then possible to find a dominating set of the given circle graph within a factor-2 approximation.

If a chord c_1 intersects with a chord c_2, then c_1 is said to *cut* c_2. This leads to a natural definition of domination within sets of chords.

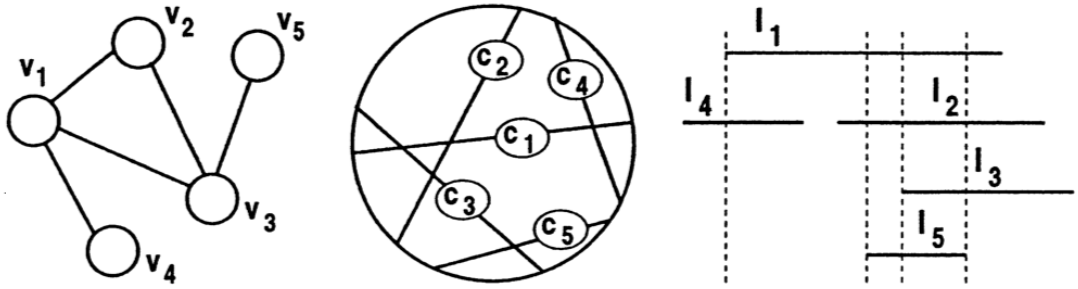

Figure 4: (a) A circle graph G (b) A chord intersection model of G (c) An interval model of G. Figure found in [18]

Definition 6.4 *Given two sets of chords C_1 and C_2, C_1* ***dominates*** *C_2 if every chord in C_2 is cut by a chord in C_1. [18]*

We define the *left endpoint l(c)* of a chord c as the endpoint with the smaller label, and the *right endpoint r(c)* as the endpoint with the larger label. The set $P(C)$ contains the set of all the endpoints of the chord set C. A similar definition of left and right holds for sectors.

For any points i, j on the circumference of a circle, the interval $(i \dots j)$ refers to the arc that is generated by tracing the circle counterclockwise from i to j. Note that this interval is open, and so i and j are not included in the arc. If it is necessary to include i and j we will refer to the arc as $[i \dots j]$.

A sequence of sectors which has the property that the $i-th$ sector's right endpoint is the $(i+1)-th$ sector's left endpoint is called a *chain of sectors*. We call the left endpoint of the first sector in a chain the left endpoint of that chain. Similarly, we call the right endpoint of the last sector in a chain the right endpoint of that chain.

Given a sector s, the set I_s denotes the chords that have at least one endpoint lying in s. The set O_s contains all of the chords whose endpoints both lie outside of s.

Finally, we can define a *leaf sector*.

Definition 6.5 *Given a sector s, if O_s dominates I_s, then s is a* ***leaf sector****. [18]*

6.1.2 Leaf Sector Cover

A *leaf sector cover* is a chain of leaf sectors C with the property that $(l(C)\ldots r(C))$ is also a leaf sector. [18]

Damian-Iordache and Pemmaraju link this problem and the minimum dominating set together by proving that for any dominating set of size d, there is a leaf sector cover of size $2d$. They then find that they can find the optimal leaf sector cover in $O(n^2)$ time. They note an important structural property of leaf sectors: if given two leaf sectors s_1 and s_2, $I_{s_1 s_2}$ can be cut with at most two chords from $O_{s_1 s_2}$. This is used to define an 8-approximation algorithm for MDS running in $O(n^2)$.

After outlining the 8-approximation algorithm they tighten the approximation factor to $2 + \epsilon$ by relaxing the notion of a leaf sector s such that I_s can be dominated by $O_s \cup (H \subset I_s)$, where $|H| = k$, k is a non-negative integer constant, and H is the smallest subset of I_s that allows O_s to dominate I_s.

6.2 Minimum Weight and Cardinality Dominating Set on Permutation Graphs

Let $G = (V, E)$ be a graph. G is a permutation graph if there is a permutation π on the set $S = \{1, 2, \ldots |V|\}$ where two vertices $i, j \in S$ are adjacent if and only if $(i - j)(\pi^{-1}(i) - \pi^{-1}(j))$ is less than 0. [19, 20]

K. Tsai and W. Hsu present algorithms for the *minimum cardinality dominating set* problem and the *minimum weight dominating set* problem on permutation graphs in $O(log\ logn)$ and $O(n^2 \cdot log^2 n)$ respectively. [20] We will cover the cardinality case.

The *minimum cardinality dominating set* problem asks for the smallest cardinality dominating set that also has the element with the largest label in it. [20]

More concretely, given $i, j \in \{1, 2, 3, \ldots, n\}$, we construct a vertex set V_i which contains the outputs of the permutation π for each i. Then we let V_{ij} be $V_i \cup \{1, 2, \ldots, j\}$. Let S be any subset of the n vertices of G. Then our goal is to construct a set D_{ij} such that D_{ij} forms a minumum cardinality dominating set. [19, 20]

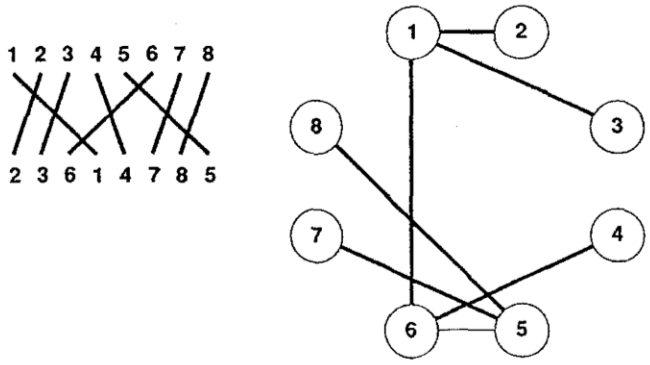

Figure 5: (a) A permutation of a set (b) The corresponding permutation graph. Figure from [20]

Tsai and Hsu improve on the work of Farber and Kiel by presenting an $O(loglogn)$ algorithm. [20] Farber and Kiel's original algorithm runs in $O(n^2)$.

Tsai and Hsu's algorithm uses dynamic programming. The improvements in their algorithm largely come from noticing a monotonicity constraint on the entries of the matrix used for the dynamic programming method. The matrix M's entries consist of $(d_{i,j}, m_{i,j})$ pairs where $d_{i,j}$ refers to the cardinality of the dominating set $D_{i,j}$, and $m_{i,j}$ refers to the largest value in that dominating set.

The structure of this matrix gives rise to the following lemma

Lemma 6.6 *If $k < j$, then $d_{ik} \leq d_{ij}$. If $k < j$ and $d_{ik} = d_{ij}$ then $m_{ik} \geq m_{ij}$. [20]*

This implies that every entry in the matrix that have the same value for d and m are contiguous, thus allowing Tsai and Hsu to create a very compact, very efficient data structure (a *block structure*) for working with the columns of the matrix. All that is required to keep track of these structures is the j-index of the highest entry in the block (the *marker* of the block). By storing the markers in an efficient priority queue, it is possible to attain a running-time of $O(log\ logn)$.

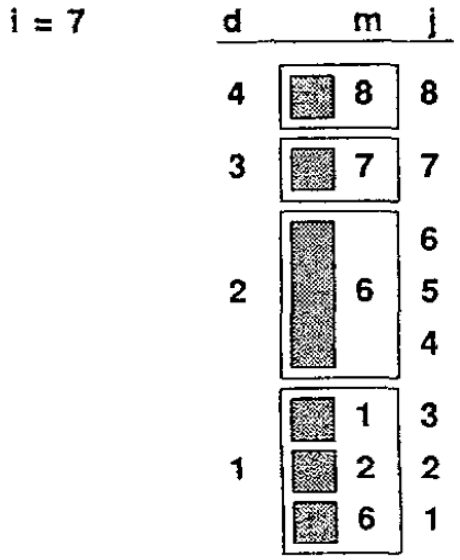

Figure 6: A block structure for a column. Image found in [20]

6.3 r-Dominating Set on Trees

X. He and Y. Yesha developed an efficient parallel algorithm for the *r-dominating set problem* on trees running in $O(logn \cdot log\ logn)$ time given n processors. The r-dominating set problem is a variant of MDS which asks for a set $D \subseteq V(G)$ which dominates every vertex in $V(G)$ within some real distance r. The model of computation in place is *concurrent read exclusive write parallel random-access* (CREW). Hence, there is the assumption that the given machine has n identical processors and a common memory store which allows concurrent reads but not concurrent writes. [21]

He and Yesha present a sequential algorithm solving the problem using a bottom-up search of the tree which labels the leaves of the tree as it progresses. They then use the fact that the minimum $r-$dominating set of the tree is determined by the final labels at the end of that algorithm.

In order to parallelize the algorithm, they translate the computation to an *algebraic computation on a binary computation tree.* A *binary computation tree* on a closed real interval S is a binary

tree whose leaves are labelled with integers from S, internal nodes are labelled with a function $f : S \times S \to S$, and whose edges are labelled with a function $g : S \to S$. The *algebraic computation* on a binary computation tree is the method used to label internal nodes of the tree.

After transforming the problem, they use a parallel tree contraction scheme to solve the algebraic problem. [21]

7 CHESSBOARDS

A *covering problem* is often used to refer to dominating sets within the scope of chess-related problems. Watkins summarizes some of these in his book "Across the Board: The Mathematics of Chess Problems". Summarized below are some of the results that Watkins placed in his book. [22]

7.1 Queens Domination

Yaglom and Yaglom show that it takes 5 queens to dominate an 8 by 8 chessboard, and that there are 4860 ways to arrange the queens that cover the chessboard. The 8 by 8 chessboard's dominating sets are interesting in that there are several coverings that place the five queens across the diagonal of the board. This raises the notion of a *queen's diagonal domination number*, which defines the minimum number of queens that must be placed along the main diagonal of an n by n chessboard. E. Cockayne and S. Hedetniemi characterize the queen's diagonal domination number in 'On the diagonal queens domination problem', and show that as the n grows large, the number of queens necessary to cover the chessboard using only the diagonal becomes greater than the number of queens sufficient to cover the board.

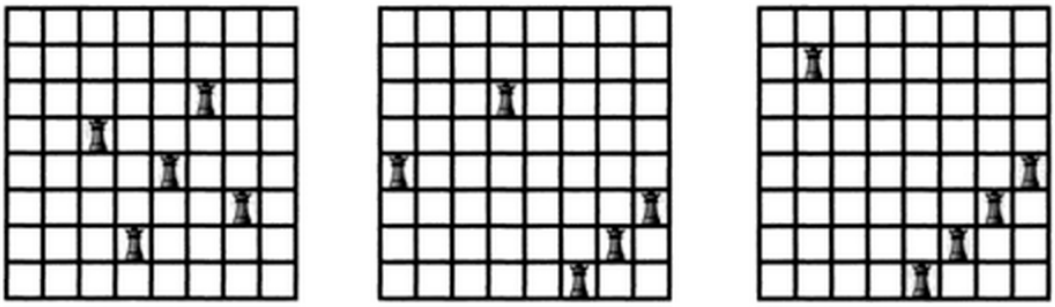

Figure 7: Some coverings of the 8 by 8 board. Figure found in [22]

The Spencer-Cockayne construction is shown to generate a covering for the chessboard, but these coverings are not always optimal. A theorem by Welch shows that the upper-bound for the number of queens necessary to cover an n by n board given $n = 3m + r$ and $0 \leq r < 3$ is

$$\gamma(G) \leq 2m + r$$

Spencer later produces what Watkins affectionately calls *Spencer's remarkable lower bound*. This states that for any n by n queens graph,

$$\gamma(G) \geq \frac{1}{2}(n-1)$$

Weakly states that if Spencer's lower bound is attained for some n by n board, that n is congruent to 3 modulo 4.

7.2 Knights Domination

Knight's domination reveals some attractive symmetries with respect to the coverings of the board, up to the point of the 11 by 11 chessboard. Although knight covering is still a hard problem, there is a linear-time algorithm due to E. Hare an S. Hedetniemi that finds $\gamma(G)$ for rectangular boards. Despite the high levels of symmetry in the lower-numbered chessboards, the optimal coverings for

the 9 by 9 and 10 by 10 chessboards feature independent knights, while the optimal covering for the 7 by 7 board consists of knights which 'guard' each other. [22]

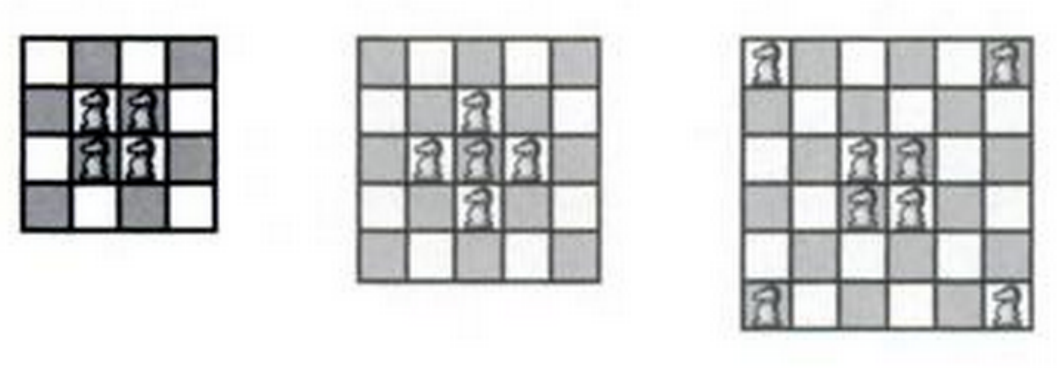

Figure 8: Some knight covers displaying symmetry. Found in [22]

7.3 Rooks Domination, Bishops Domination

The domination number for a rook's graph is known to be $\gamma(G) = min(n, m)$. The domination number for a bishop's graph of size n by n is $\gamma(G) = n$.

8 DOMINATING HAMMING GRAPHS

The Information System on Graph Classes and their Inclusions (ISGCI) [23] states that there is currently no data on the hardness of solving dominating set problems on Hamming graphs.

A *Hamming graph* is a graph that arises from the notion of the *Hamming distance* of two vectors. Hamming graphs have vector-labelled vertices which have a distance between them equal to the Hamming distance between their labels. [23]

An interesting subcase of the Hamming Graphs is the *rook's graph.* [24] The rook's graph is a representation of the legal moves a rook may make on an n by m chessboard. Although it seems that there is not much data on general Hamming graphs with respect to dominating set

problems, chessboard-related problems lend themselves well to domination. There is a lower-bound of $min(n, m)$ on $\gamma(G)$ for a rook's graph G. [22]

An alternative definition of Hamming graph also provided by the ISGCI is that a (complete) Hamming graph is the Cartesian product of d complete graphs, which may or may not be isomorphic. S. Gravier presents some bounds on the minimum total dominating number of Cartesian products of paths and cycles, but nothing on complete graphs. [25]

REFERENCES

[1] J. Barilan, G. Kortsarz, and D. Peleg, "How to allocate network centers," *Journal of Algorithms*, vol. 15, no. 3, pp. 385 – 415, 1993. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0196677483710473

[2] T. Nieberg, J. Hurink, and W. Kern, "Approximation schemes for wireless networks," *ACM Trans. Algorithms*, vol. 4, no. 4, pp. 49:1–49:17, Aug. 2008, quality A*. [Online]. Available: http://doi.acm.org.ezproxy.library.uvic.ca/10.1145/1383369.1383380

[3] J. Wu, F. Dai, and S. Yang, "Iterative local solutions for connected dominating set in ad hoc wireless networks," *Computers, IEEE Transactions on*, vol. 57, no. 5, pp. 702–715, May 2008, quality A*.

[4] C. Martinez, R. Beivide, and E. Gabidulin, "Perfect codes for metrics induced by circulant graphs," *Information Theory, IEEE Transactions on*, vol. 53, no. 9, pp. 3042–3052, Sept 2007, quality A*.

[5] F. Wang, H. Du, E. Camacho, K. Xu, W. Lee, Y. Shi, and S. Shan, "On positive influence dominating sets in social networks," *Theoretical Computer Science*, vol. 412, no. 3, pp. 265 – 269, 2011, combinatorial Optimization and Applications {COCOA} 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397509007221

[6] F. V. Fomin and P. Kaski, "Exact exponential algorithms," *Commun. ACM*, vol. 56, no. 3, pp. 80–88, Mar. 2013, quality A*. [Online]. Available: http://doi.acm.org.ezproxy.library.uvic.ca/10.1145/2428556.2428575

[7] F. Fomin, D. Kratsch, and G. Woeginger, "Exact (exponential) algorithms for the dominating set problem," in *Graph-Theoretic Concepts in Computer Science*, ser. Lecture Notes in Computer Science, J. Hromkovi, M. Nagl, and B. Westfechtel, Eds. Springer Berlin Heidelberg, 2005, vol. 3353, pp. 245–256. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30559-0_21

[8] B. Reed, "Paths, stars and the number three," *Combinatorics, Probability*

and Computing, vol. 5, pp. 277–295, 9 1996, quality A. [Online]. Available: http://journals.cambridge.org/article_S0963548300002042

[9] F. V. Fomin, F. Grandoni, and D. Kratsch, "A measure & conquer approach for the analysis of exact algorithms," *J. ACM*, vol. 56, no. 5, pp. 25:1–25:32, Aug. 2009, quality A*. [Online]. Available: http://doi.acm.org.ezproxy.library.uvic.ca/10.1145/1552285.1552286

[10] F. Fomin, F. Grandoni, and D. Kratsch, "Solving connected dominating set faster than 2^n," *Algorithmica*, vol. 52, no. 2, pp. 153–166, 2008. [Online]. Available: http://dx.doi.org/10.1007/s00453-007-9145-z

[11] E. Camby and O. Schaudt, "The price of connectivity for dominating set: Upper bounds and complexity," *Discrete Applied Mathematics*, vol. 177, no. 0, pp. 53 – 59, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166218X14002443

[12] J. Hromkovic and W. M. Oliva, *Algorithmics for Hard Problems*, 2nd ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2002, book, unranked.

[13] G. Philip, V. Raman, and S. Sikdar, "Polynomial kernels for dominating set in graphs of bounded degeneracy and beyond," *ACM Trans. Algorithms*, vol. 9, no. 1, pp. 11:1–11:23, Dec. 2012, quality A*. [Online]. Available: http://doi.acm.org.ezproxy.library.uvic.ca/10.1145/2390176.2390187

[14] J. Ellis, H. Fan, and M. Fellows, "The dominating set problem is fixed parameter tractable for graphs of bounded genus," *Journal of Algorithms*, vol. 52, no. 2, pp. 152 – 168, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0196677404000409

[15] V. Kann, "On the hardness of np-complete optimization problems," dissertation, 1992. [Online]. Available: http://www.csc.kth.se/ viggo/papers/phdthesis.pdf

[16] L. A. Sanchis, "Experimental analysis of heuristic algorithms for the dominating set problem," *Algorithmica*, vol. 33, no. 1, pp. 3–18, 2002, quality A*. [Online]. Available: http://dx.doi.org/10.1007/s00453-001-0101-z

[17] M. Damian-Iordache and S. V. Pemmaraju, "A $(2+\epsilon)$-approximation scheme for minimum domination on circle graphs," *Journal of Algorithms*, vol. 42, no. 2, pp. 255 – 276, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0196677401912064

[18] M. Damian and S. V. Pemmaraju, "Apx-hardness of domination problems in circle graphs," *Information Processing Letters*, vol. 97, no. 6, pp. 231 – 237, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020019005003224

[19] M. Farber and J. M. Keil, "Domination in permutation graphs," *Journal of Algorithms*, vol. 6, no. 3, pp. 309 – 321, 1985. [Online]. Available: http://www.sciencedirect.com/science/article/pii/019667748590001X

[20] K.-H. Tsai and W.-L. Hsu, "Fast algorithms for the dominating set problem on permutation graphs," *Algorithmica*, vol. 9, no. 6, pp. 601–614, 1993, quality A*. [Online]. Available: http://dx.doi.org/10.1007/BF01190158

[21] X. He and Y. Yesha, "Efficient parallel algorithms for r-dominating set and p-center problems on trees," *Algorithmica*, vol. 5, no. 1-4, pp. 129–145, 1990. [Online]. Available: http://dx.doi.org/10.1007/BF01840381

[22] J. J. Watkins, *Across the Board: The Mathematics of Chessboard Problems*. Princeton University Press, 2012, book, unranked.

[23] H. N. de Ridder *et al.*, "Information System on Graph Classes and their Inclusions (ISGCI)," http://www.graphclasses.org, unranked.

[24] R. F. Bailey and P. J. Cameron, "Base size, metric dimension and other invariants of groups and graphs," *Bulletin of the London Mathematical Society*, vol. 43, no. 2, pp. 209–242, 2011, unranked. [Online]. Available: http://blms.oxfordjournals.org/content/43/2/209.1.abstract

[25] S. Gravier, "Total domination number of grid graphs," *Discrete Applied Mathematics*, vol. 121, no. 13, pp. 119 – 128, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166218X01002979