An Introduction to Integrated Domination

Using K_1's and K_2's

By

Jennie J. Newman

A Thesis Submitted to
Saint Mary's University, Halifax, Nova Scotia
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science (Honours).

April, 2016, Halifax, Nova Scotia

Approved:	Dr. Bert Hartnell Supervisor
Approved:	Dr. Art Finbow Reader
Date:	May 16, 2016

An Introduction to Integrated Domination Using K_1's and K_2's

by Jennie J. Newman

Abstract

In Mathematics, graph theory is the study of graphs, which consist of a set of points, called vertices, and the connections between them, called edges. Domination is a subfield of this study, which looks at subsets of vertices in a graph that are adjacent to every other vertex in the graph. These subsets are called dominating sets. A vertex u is said to dominate a vertex v, if u is adjacent to v.

The principal problem of domination is to find the smallest dominating set for a graph. Variations of domination exist, with the two standard types using K_1's (single vertices), or K_2's (paired vertices) as guards.

To the best of our knowledge, we introduce the idea of integrating two different types of guards in one dominating set. Here, we look at the idea of dominating a graph using a combination of guards in the forms of K_1's and K_2's, and the problem of finding a minimum dominating set for this style of domination, which we call *integrated domination*.

We look at a number of well-known variations of domination, and then characterize graphs and subgraphs where a minimum integrated dominating set can efficiently be found. As well, we present a bound for this style of domination, and then discuss further directions for this problem.

April, 2016

Acknowledgements

First and foremost, I'd like to thank Dr. Bert Hartnell. In my second year at Saint Mary's University he introduced me to the topic of graph theory, and opened my eyes to the diverse world of Math.

Since that time he has helped me in countless ways, with school and life, and has encouraged me and inspired me to work hard and strive towards my passions. The thing I must thank him for the most though, is believing in me. The last few years would not have been the same if it weren't for him.

I would also like to thank Rose Daurie, as well as all of the professors at Saint Mary's University who have also greatly helped me and encouraged me over the years. I hold dear the advice and time you have given me, and I would not be where I am today without each of you.

Finally, I'd like to thank my family for all their support along the way.

Contents

List of Tables

List of Figures

Introduction

Mathematical work regarding domination in graphs is believed to have began in the 1950's, coming from mathematicians such as Kõnig (1950) [23], and Ore (1962) [28]. The topic itself, however, dates back to the 1800's, when a European man named de Jaenisch thought about the number of queen pieces required to cover, or dominate, an n x n chessboard [9].

Since that time, the topic has increased significantly in popularity, which many believe to be due to its relation to real-world and theoretical locating and covering problems, the diversity in the parameters that can be used for domination, and due to the general domination problem being determined as NP-complete [20].

The problem of domination deals with the idea of finding subsets of vertices in a graph, that are adjacent to all other vertices in the graph. In many works, the vertices in the dominating set D, can be called guards, and each guard can dominate itself plus its adjacent vertices. The main problem looked at with domination, regardless of the variation, is that one wants to find the minimum number of guards necessary in order to protect the entire graph.

In regards to real world application, this problem can be related to protecting a computer network from intruders, guarding corridors in a building, ensuring towns have enough resources, and more. Figure 0.1 illustrates a few of these ideas.

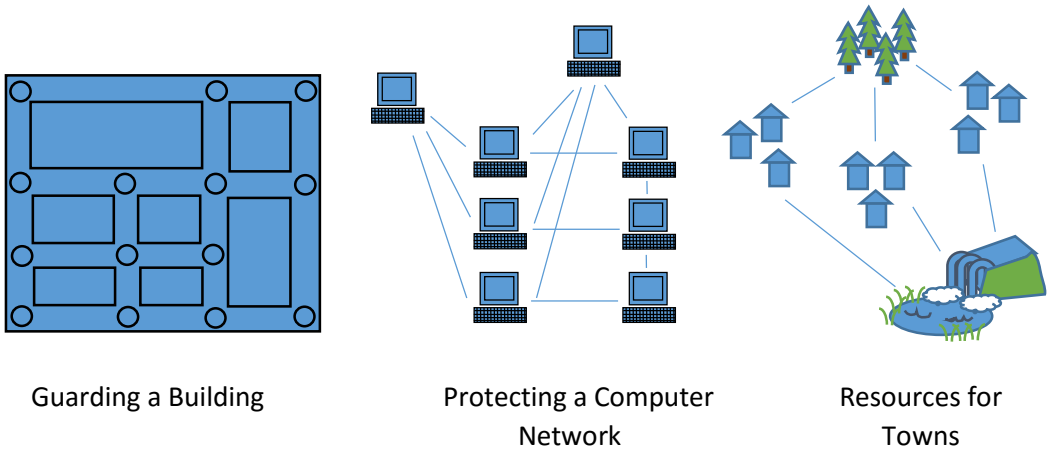

Figure 0.1: An illustration of real world applications.

As of today, there are many variations and styles of domination where restrictions are placed on the types of guards used, the number of vertices allowed to be dominated by a guard, the positioning of the guards, and others. We discuss a number of these styles of domination in the following section, and some of their results.

In this work we introduce, to the best of our knowledge, a new form of dominating a graph where we use a combination of two different guard types. We look at dominating a graph using guards in the forms of K_1's and K_2's, which we define later, and the problem of finding a minimum dominating set for this style of domination, which we call *integrated domination*. Topics such as graphs and subgraphs where a minimum integrated dominating set can efficiently be found will be examined, as well as bounds for this style of domination.

Definitions

A *vertex*, or *node*, is a single point, and an *edge* is a connection between two vertices, often labeled as an unordered pair of two vertices. A *graph* G = (V, E) is a set of vertices and edges, with V representing the set of vertices, and E representing the set of edges. The *degree* of a vertex v is the number of edges connected to v, and is usually denoted as deg(v). Two vertices are said to be *adjacent* if they share an edge. A vertex u is said to be a *neighbour* of a vertex v, if u is adjacent to v.

The *neighbourhood* of a vertex v is the set of vertices adjacent to v, and is denoted N(v). A *closed neighbourhood* includes v in this set, and an *open neighbourhood* excludes v. A *partition* of a graph G, is a collection of disjoint subsets whose union equals G. A set of vertices is said to be *independent* if no two vertices in the set are adjacent.

The *cardinality* of a set is the number of items in the set, and is denoted |S|. For example, the set $\{v_1, v_2, v_3, v_4\}$ has a cardinality of 4. A set in a graph is said to be *minimal* if the removal of an element in the set destroys the given property of the set. A set is said to be *minimum* if it is the smallest of all the minimal sets. We can say that minimum is always minimal, however minimal is not always minimum. Figure 0.2 illustrates this concept, where the property of the set is that it is dominating.

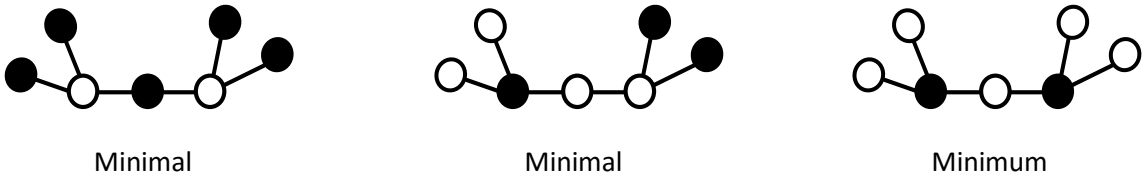

Figure 0.2: Minimal and minimum sets.

A *leaf*, or *endvertex*, is a vertex of degree 1. A *stem* is a vertex with at least one leaf neighbour. A *path* is a string of adjacent vertices $\{v_1, v_2, \ldots, v_n\}$ such that v_1 and v_n are leaves, and every other vertex is of degree 2. A *cycle* is a path such that $v_1 = v_n$. In this work we denote a *high degree vertex* to be a vertex of degree ≥ 3, unless otherwise stated.

A path of vertices connected to a high degree vertex, such that each vertex in the path, other than the endvertex, is of degree 2, will be called a *k-arm*, with k being the number of vertices in the path excluding the high degree vertex. A path of k degree 2 vertices joining two stems will be called a *k-path*. Given a cut-vertex x in a graph G, each component of G\{x] is called a *branch* at x. A *complete graph* or a *clique* is a graph in which each vertex has an edge between itself and every other vertex in the graph. It's usually denoted as K_n, with n being the number of vertices in the graph. The *length* of a path or k-arm refers to the number of distinct vertices in the path, or on the k-arm excluding the high degree vertex.

A *corona* of two graphs $G \circ H$, is the graph formed from one copy of G and k copies of H, where k is the number of vertices in G. The copies are connected so that the i*th* vertex of G is adjacent to all vertices in the i*th* copy of H. A graph G is *chordal* if every cycle of

length ≥ 3 in G has an edge between two non-consecutive vertices in the cycle. G is *strongly chordal* if it is chordal and every cycle of even length ≥ 6 has an edge connecting two non-consecutive vertices whose distance on the cycle is of odd length.

A set S of vertices in G is a *dominating set* if every vertex in G is either in S or adjacent to a vertex in S. A vertex which dominates another vertex may be called a *dominating vertex*, or a *guard*. If S is a dominating set with T being a K_1 or K_2 in S, and if v is a vertex that T lies on, then we say that v *hosts* the guard T. The *domination number* $\gamma(G)$, is the cardinality of a minimum dominating set of a graph G. Such a set in a graph is said to have *overlap* if there exists a vertex which is dominated by more than one dominating vertex.

When determining a specific value, a *lower bound* gives an estimate such that the value will be no smaller than the estimate. Similarly, an *upper bound* gives an estimate such that the value will be no larger than the estimate.

Definitions Regarding the Computational Complexity of Problems

An *algorithm* is a set of steps followed in order to solve a problem.

The standard way of expressing the speed or *computational complexity* of an algorithm is in the form of a function f(n), with n being the size of the input, and the value of f(n) being the number of steps required to perform the algorithm. For example, n could represent the number of vertices of a graph, or the number of items needing to be sorted. An *efficient algorithm* is an algorithm with the speed represented as a linear, polynomial,

or logarithmic function. Exponential functions such as 2^n and n! are not considered efficient, as they use more steps as n becomes larger.

Regarding the computational complexity of algorithms, a problem of interest among computer scientists deals with NP-completeness, and asks whether P=NP (we will understand these terms soon). NP-completeness deals with sets of problems classified by their computational complexity. To define NP-completeness, we first must look at a few other defintions.

A *decision problem* is a problem or question whose answer is a yes or no (accepted or rejected) depending on the input. A *deterministic algorithm* is an algorithm in which the same number of steps is taken each time the algorithm is performed, and each time the same result is given for the same set of data. A *nondeterministic algorithm* is an algorithm which "guesses" the next step to take, and thus it can vary the way it performs, i.e., it may use a different number of steps and return different results even for the same set of data, or it may never finish.

P refers to the set of problems that can be solved using a deterministic algorithm in polynomial time. *NP* refers to the set of problems where a solution can be generated from a nondeterminisitic algorithm, and where this solution can be verified or rejected in polynomial time.

A set is *polynomial-time reducible* to another set if there exists a polynomial function f(x) that maps every element in the first set to the second set, and vice versa, formally: for all x, x is in the first set if and only if f(x) is in the second set.

In terms of problems, a problem is said to be polynomial-time reducible to another problem if its inputs can be transformed to the inputs of the second problem. Generally, showing that a problem A is polynomial time reducible to problem B means problem B is at least as hard as problem A, since if problem B can be solved, then problem A can also be solved.

Now, a decision problem is said to be *NP-complete* if the problem falls into the NP set, and if any problem in the NP set can be polynomial-time reduced to this decision problem. A decision problem is said to be *NP-hard* if it satisifies the second condition mentioned above.

Applied Combinatorics, 5th Edition; *Computational Complexity: A Conceptual Perspective*; *Fundamentals of Domination in Graphs*; and *Review: Michael R. Garey and David S. Johnson, Computers and intractability: A guide to the theory of NP-completeness*, were used as references for this information, and can be referenced for further information on this topic [35, 15, 17, 1].

In this work we will see that many problems regarding domination have been determined to be NP-complete.

1 Related Works

1.1 Variations of Domination

As domination is a topic of interest to many mathematicians, we present some background information into the original form of domination, as well as some variations of the problem.

1.1.1 Domination

The standard and original type of domination uses K_1's, or single vertices, to dominate a graph, where a K_1 guard can dominate the vertex it is on, plus all adjacent nodes. Again, the problem of interest is to dominate as many nodes as possible, using the fewest number of guards, with the intent of guarding the entire graph. This problem not only involves finding the minimum number of guards needed, but also largely deals with where to place these guards. Finding a minimum dominating set for general graphs is quite hard, and was determined to be NP-complete. Many mathematicians have worked on this problem, however as of yet there is no known efficient solution for an arbitrary graph [20].

In regards to trees however, a simple linear algorithm for finding the domination number exists, and was created in 1975 by E. Cockayne, S. Goodman and S. Hedetniemi [5]. The algorithm involves rooting a tree and giving each vertex a label, as either Bound, Free or Required, where Bound are the vertices that need to be dominated, Free are those that need not be dominated, and Required are the ones that must be in the dominating set.

The algorithm starts with rooting the tree at an arbitrary point and numerically labelling the vertices, working towards the outside of the tree in any manner (e.g., a breadth-first search method would work). Each vertex is then given a label as either Bound, Free or Required. We note that the parent of a vertex *i*, is its neighbour with the smallest number label. Working through the graph in descending numerical order, each vertex's label is compared to its parent's and is possibly relabelled, according to Table 1.1. After checking, if a vertex is labelled as Required, a K_1 is placed on it. When the final vertex (the root, [1]) is reached, if it is labelled as Bound or Required, a K_1 is placed on it as well, and the minimum dominating set is complete.

Figure 1.1 illustrates the process of guarding an arbitrary tree using this algorithm, where every vertex starts with a Bound label.

Table 1.1: Relationship between parent and child vertices for a domination algorithm.

Parent				
Child		**Free**	**Bound**	**Required**
	Free	Do nothing	Do nothing	Do nothing
	Bound	Required	Required	Do nothing
	Required	Do nothing	Free	Do nothing

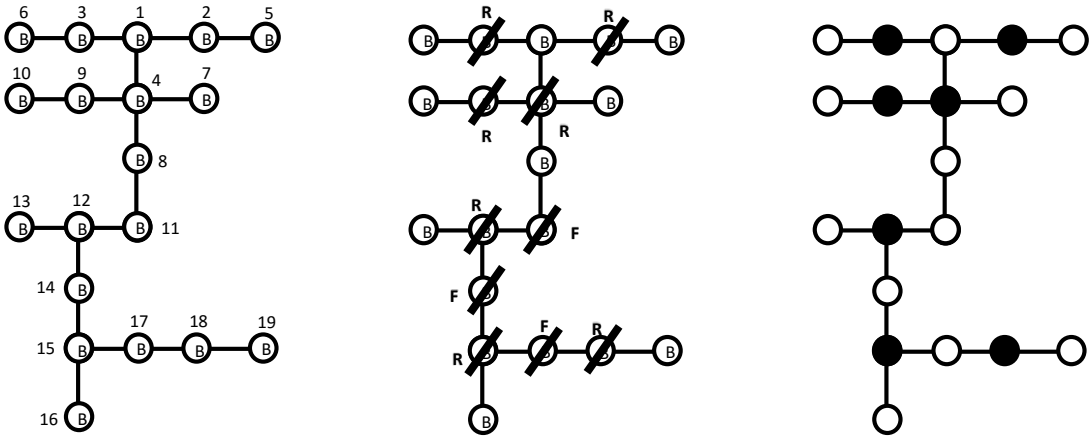

Figure 1.1: An example of the domination algorithm.

Having each vertex start with a Bound label is quite practical, especially for situations where a new graph is given and a dominating set needs to be found. This is the case for our problem as well. However, for situations where some guards are already installed in a graph, but more guards are needed, this algorithm has the ability to help here as well. For these, one would label the vertices with guards as Required, and all others as Bound or Free, depending on if they needed to be guarded.

In terms of real world application, this could relate to having facilities (fire stations, police stations, etc.) already established in a town, yet needing more to accommodate a growing population or town size, and so one would want to find the optimal location for these. It can also relate to guarding a building, with some rooms empty and not needing guarding, and some with items of importance in them and so requiring a guard in them at all times.

This algorithm is very useful for trees and now is very well-known. Although, as of yet, there is no efficient algorithm to find the minimum dominating set for graphs in general, theorems have been discovered to give more insight into the general characteristics of dominating sets in graphs.

To start, from one of the first mathematicians to look at domination in graphs, Ore, we have the following theorem: If a graph G, has no isolated vertices then $\gamma(G) \leq n/2$, where n is the number of vertices in the graph [28].

Another well-known theorem is: Every non-trivial connected graph G has a dominating set D, whose complement V\D is also a dominating set [25].

Going further in this way: If G is a graph without isolated vertices, for any minimal dominating set D, its complement V\D is also a dominating set [25].

These theorems have nice proofs, with the third of such proceeding as follows: Say D is a minimal dominating set in an arbitrary such graph, G. By way of contradiction, one assumes the set V\D is not a dominating set. This means that there exists a vertex u in G, such that u is not adjacent to any of the vertices in V\D. It also means that u must be dominated by at least one vertex in D\{u}, as there are no isolated vertices in the graph. Because of this, the set D\{u} must also be a dominating set. However, this contradicts the fact that D is a minimal dominating set, and thus shows that V\D must also be a dominating set.

As a corollary to this statement, it is also noted that the domination number of a graph without isolated vertices is less than or equal to half of the number of vertices in the graph, stated formally as $\gamma(G) \leq \frac{1}{2}|V|$. The proof follows from the previous theorem, where if D is a minimal dominating set, and V\D is a dominating set, the vertices in D and those in V\D must together equal the total number of vertices in G. Thus, if one of these sets contains more than half of the vertices of V, then the other set will contain less than half.

The original form of domination has some nice and convenient theorems, which form a basis for further study of domination. Next we look at the second standard type of domination; paired-domination.

1.1.2 Paired-Domination

The idea of paired-domination was first introduced by T. Haynes and P. Slater in 1995 [18]. Closely related to the standard domination mentioned previously, paired-domination uses two adjacent K_1's as a guard, rather than one single K_1. Here, both K_1's guard the vertices they are on, plus any adjacent vertices. The idea behind paired-domination was that if a situation happens where one guard becomes unavailable, another guard is right there for backup.

Another way to think of this problem is to use K_2's as guards, rather than two K_1's. This can relate to the idea of one guard moving back and forth between two fixed points. In this dynamic style, the guard protects the two nodes it visits plus any adjacent nodes to those two.

Similar to the issue with standard domination, the problem of finding a minimum paired-dominating set for a general graph is quite difficult and was determined to be NP-complete by Haynes and Slater themselves. Although no efficient solution exists in general as of yet, the two presented some bounds for the problem, stating: If a graph G has no isolated vertices, then the paired-domination number, $\gamma_{pr}(G)$ will be at least 2, and at most n, where n is equal to the number of vertices in G, and these bounds are sharp [18].

More specifically, they showed that if a graph G has no isolated vertices, then the paired-domination number of G will be at most two times the domination number of G i.e., $\gamma_{pr}(G) \leq 2\ \gamma\ (G)$. This stems from the idea that for a dominating set S in a graph G, if one pairs each vertex in S with one of its private neighbours to achieve paired-domination, then

the result will be double the domination number. The bound is in fact tight, as illustrated by P_6, a path on 6 nodes, as shown in Figure 1.2 [18].

Figure 1.2: Dominating and paired-dominating sets for a P_6.

Nevertheless, an efficient algorithm exists for solving this problem for trees, and was recently discovered by H. Qiao, L. Kang, M. Cardei, and D. Du in 2003 [31].

Interestingly, the problem of paired-domination is also closely related to total domination, which we look at next. Figure 1.3 shows two examples of paired-domination in graphs.

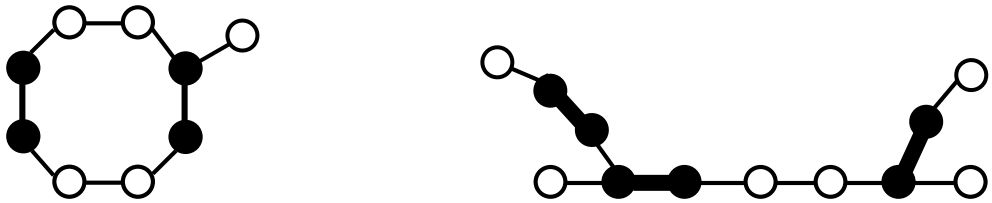

Figure 1.3: Examples of paired-domination.

1.1.3 Total Domination

In paired-domination, the guards are found in pairs and so dominate each other, resulting in every vertex being dominated by a guard other than itself. This is precisely the idea behind total domination, where a vertex can only dominate its open neighbourhood, and

not itself. The difference between these two though, is that for total domination the guards are not required to be paired.

Total domination was inspired by a solution to a well-known problem from Europe in the 1850's; the Five Queens Problem, as mentioned earlier. The problem, chess-related, deals with an 8 x 8 square chess board and the Queen piece. The problem asks the minimum number of queens needed in order to ensure that every square on the board can be attacked or occupied by a queen, where a queen can move any number of squares in any one direction. The solution, determined to be five, not only showed that every square was dominated by a queen at one time, but also showed that each queen was being dominated by another. It was this observation that led to the start of total domination.

Mathematicians Cockayne, Dawes, and Hedetniemi formally introduced this concept in 1980 [3]. Real-world applications of the problem can relate to computer networks, where a main group of computers can, as a whole, connect with every other computer in the network, but also where each main computer is connected to another main, for backup purposes. Similarly, it can relate to committee selection, where everyone outside the committee knows someone on the committee, and each committee member knows another member on the committee [3].

In terms of finding a total dominating set for an arbitrary graph, this was shown to be NP-complete by mathematician J. Pfaff, in 1984 [29].

However, finding a total dominating set for an arbitrary tree can be done in polynomial time, thanks to a linear algorithm created by R. Laskar, J. Pfaff, S. Hedetniemi, and S. Hedetniemi in 1984 [26].

Moreover, there exist theorems to characterize and give bounds for the total domination number in a general graph, $\gamma_t(G)$. For example, a total dominating set S, is minimal if and only if the set $OB(S) = \{v \mid |N(v) \cap S| = 1\}$ dominates S, where OB(S) is the set of vertices such that the intersection of the neighbourhood with these vertices and S contain only one vertex [19]. Further, if a graph is connected and has at least 3 vertices, then $\gamma_t(G) \leq 2n/3$ [3].

Figure 1.4 shows an example of a solution to the Five Queens problem [17].

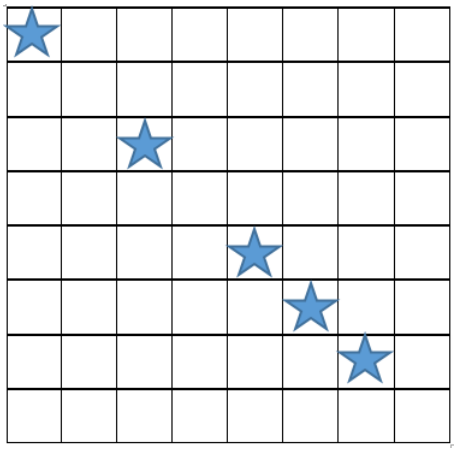

Figure 1.4: A total dominating set solution to the Five Queens problem.

1.1.4 Dominating Cliques

Moving on to another form of domination, in 1990 M. Cozzens and L. Kelleher introduced the idea of dominating cliques, in which the vertices of the dominating set must induce a complete graph [8]. In relation to the real world, this problem can relate to guarding computer networks, with each node representing a computer, and each edge representing

the connection between two computers. The clique represents a group of principal computers where each computer in the group can communicate with each other, and as a whole, communicate with every other computer in the network. Similarly, Cozzens and Kelleher also worked on this idea in social networks where a person is represented by a vertex, and the connection between two people is represented by an edge [22].

Resulting from their work, Cozzens and Kelleher found a condition for graphs with dominating cliques, stating that if G is a connected graph with no induced P_5 or C_5, then G has a dominating clique. They also established an upper bound for the clique domination number $\gamma_{cl}(G)$, which says if G is a connected graph with no induced P_5, C_5, or corona $K_{k+1} \circ K_1$, then $\gamma_{cl}(G) \leq k$.

For connected graphs with no induced P_5 or C_5, Cozzens and Kelleher also created an efficient, polynomial time, algorithm to find a dominating clique. D. Kratsch also presented a polynomial time algorithm to solve the problem for certain types of graphs, including strongly chordal graphs. However, the decision problem regarding finding a dominating clique for general graphs is NP-complete [24].

Figure 1.5 gives two examples, where a K_5 and a K_4 are used as dominating cliques.

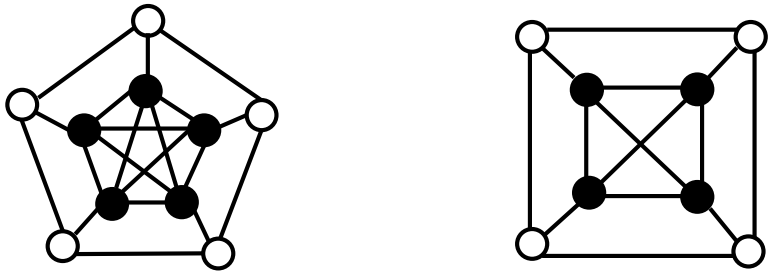

Figure 1.5: Examples of dominating cliques

1.1.5 Multiple Domination

Of course, other variations of domination in graphs exist, with the altered conditions being on the types of guards used, the requirements and positions of the guards, and more. One example that focuses on the positioning of the guards is multiple domination, which says that each vertex in V\D must be dominated by a specific number k, of vertices. This idea was introduced by J. Fink and M. Jacobson in 1985 [13]. These two discovered certain bounds for this type of domination, such as $\gamma_k(G) \geq kn/(\Delta(G) + k)$ which states that for a graph G, the k-domination number $\gamma_k(G)$, will be greater than or equal to k times n, divided by the sum of the maximum degree of a vertex in G $\Delta(G)$, plus k, with n being the number of vertices in the graph.

On the opposite side, they found that if a graph has $k \leq \delta(G)$, then $\gamma_k(G) \leq kn/(k+1)$, which again says that if the k value is less than the minimum degree of any vertex in G $\delta(G)$, then the k-domination number is less than or equal to k times n, divided by the sum of k and 1. This upper bound was discovered by E. Cockayne, B. Gamble, and B. Sheppard, in 1985 [4].

In Figure 1.6 we see an example of a 2-dominated graph and a 3-dominated graph.

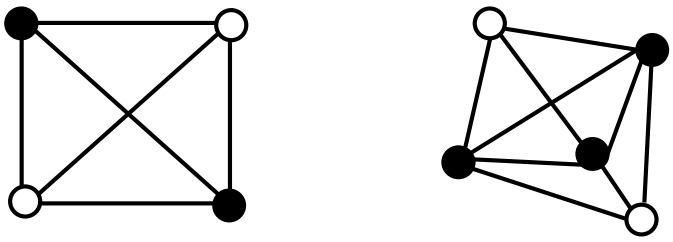

Figure 1.6: Examples of multiple domination.

1.1.6 Distance Domination

Distance domination is another variation, with the restriction being that every vertex in V\D must be within a certain distance of the vertices in D. So, a *distance-k* dominating set S, must have every vertex in V\S within a distance of k nodes of at least one vertex in S. Two simple examples are shown in Figure 1.7, where each vertex in G is within a distance of 2 nodes from any guard.

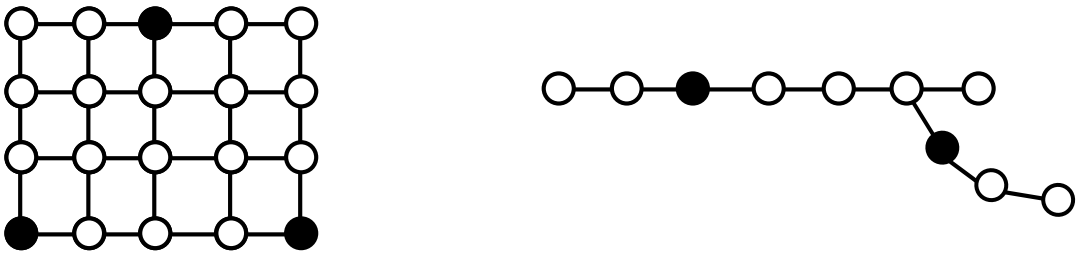

Figure 1.7: Examples of distance-2 domination.

A real world application came from the problem of placing a minimum number of objects or institutions within a reasonable distance of a certain population. Such objects or institutions could be fire stations, post offices, schools, etc. Oppositely, the problem could be in regards to placing unwanted objects as far away from a population as possible, referring to objects such as nuclear reactors, garbage plants, airports, etc.

The idea that the distance k, could vary from vertex to vertex was looked at by P. Slater in 1976 [32], and some characterizations for distance-k domination of graphs were given by M. Henning, O. Oellermann, and H. Swart in 1991 [21].

1.1.7 Others

There are a few other variations of domination that we briefly mention without going into much detail.

Parity restrictions restricts the vertices in V to being dominated by an odd number of vertices, or by a specific number of vertices only. Finding the minimum cardinality of an all-odd parity set was determined to be NP-hard by K. Sutner, in 1989 [34].

Locating domination has the restriction that for each vertex in V\D, there is a unique set of vertices that can dominate each of them. The idea stems from a nautical point of view, where if a ship runs into problems while at sea, having a system of buoys set up, if a crew member could call back to shore and communicate which buoys he saw, then the folks back on shore would be able to pinpoint his location and send help. C. Colbourn, P. Slater, and L. Stewart determined this problem to be NP-hard in 1987 [7].

Strong domination, and similarly weak domination, require that each vertex in V\D must be dominated by at least one vertex whose degree is greater than (or less than, for weak domination) the degree of itself. According to *Fundamentals of Domination in Graphs* [17], it is known but unpublished that the strong and weak domination problems are NP-complete, and that linear algorithms exist for finding the strong and weak domination numbers for arbitrary trees.

Of course there are many other forms of domination in graphs, however these few mentioned should give an idea of how domination works and why it is a problem of interest.

1.2 Domination and Other Parameters

1.2.1 Dominating, Independent, and Irredundant Sets

Another important theorem, concerning the bounds for domination in graphs, is related to independent sets and irredundant sets in graphs.

First, we note that for a dominating set, the lower bound is commonly denoted $\gamma(G)$, and refers to the minimum cardinality of all dominating sets of G. The upper bound for the set is denoted $\Gamma(G)$, and describes the largest cardinality for a minimal dominating set.

Recall that in a graph G, a set of vertices such that no vertex is adjacent to another is called an independent set, with two vertices being called independent if they are not adjacent.

A set S of vertices in a graph, such that for every vertex v in S, the neighbourhood of v contains a vertex w, where w belongs to no other neighbourhood of an element in S, is called an irredundant set of vertices. This means that for every vertex v in S, v has a private neighbour; a vertex that belongs only to its neighbourhood, and no one else's. If a vertex satisfies this condition, it's called an irredundant vertex.

Figure 1.8 illustrates these ideas.

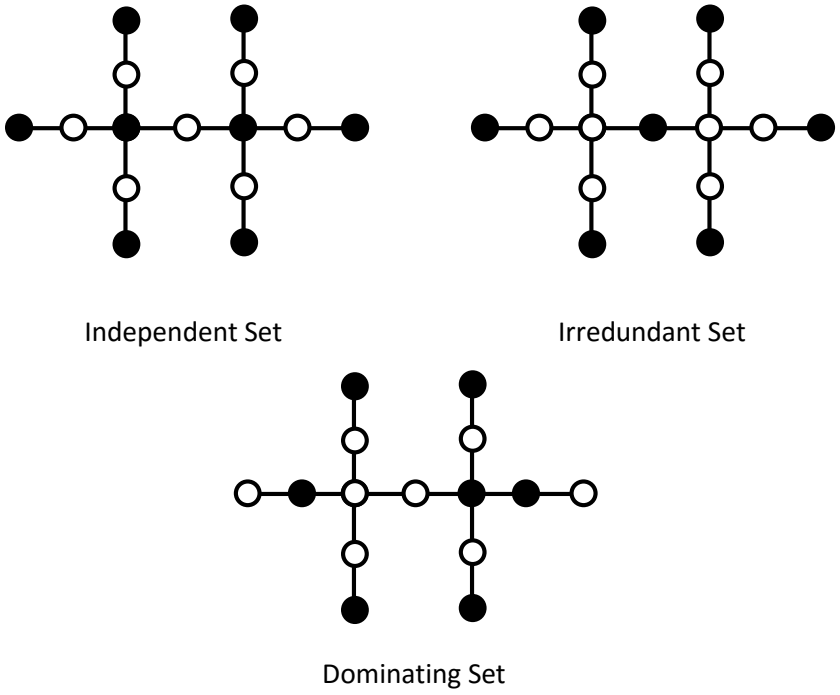

Figure 1.8: Independent, irredundant and dominating sets.

The three ideas of domination, independence and irredundance in graphs fit together nicely, with proven statements such as:

A set of vertices in a graph is an independent dominating set if and only if S is maximal independent [2].

In a graph, every maximal independent set of vertices is a minimal dominating set [2].

Every minimal dominating set of vertices is an irredundant set [2].

1.2.2 Parameters

The independent domination number is the cardinality of the minimum independent dominating set of a graph, and is denoted $i(G)$.

The independence number of a graph is denoted $\beta(G)$, and refers to the cardinality of the maximum independent set of G.

The irredundance number of a graph $ir(G)$, is the smallest cardinality of all maximal irredundant sets of G.

The upper irredundance number, conversely, is the largest cardinality of all irredundant sets of a graph, and is denoted $IR(G)$ [2].

These six parameters interlace quite nicely with the following theorem, which was first discovered by Cockayne and Hedetniemi in 1977:

For every graph G, $ir(G) \leq \gamma(G) \leq i(G) \leq \beta(G) \leq \Gamma(G) \leq IR(G)$ [6].

1.3 Some Techniques for Hard Problems

As is the case with some of the previous types of domination looked at, finding a method or algorithm for guarding a general graph may be quite difficult or NP-complete. Rather than looking for a direct solution to these problems, another approach is to ask questions such as, “For which graphs is it easy to solve?’ and “Are there approximations to the problem?”

An example of such a strategy is used in the problem of well-covered graphs. The problem, as we will discuss briefly, is relatively hard, and so what some mathematicians worked on, was finding properties of the problem that can relate to other well-known problems, and to characterize graphs which can be well-covered.

Approaching a problem in this way does not only give more information on the problem itself, but it may give insight into other problems as well. Such is the case with the well-covered problem and a problem known as well paired-domination, which we will look at soon. This will be the case for us as well, where we will see that characterizations from similar problems can also suit our problem.

1.3.1 Well-Covered

A vertex cover in a graph is a set of vertices that meets every edge in the graph. A vertex cover is said to be minimal if the removal of any vertex in the graph destroys the covering property. A well-covered graph is one in which every minimal vertex cover has the same cardinality as every other minimal vertex cover, and thus every minimal cover is also minimum.

It was M. Plummer who introduced the idea of a well-covered graph in 1970, and noted that since the complement of a vertex cover is an independent set, every maximal independent set is also maximum [30]. Since then, other mathematicians have expanded on the idea. One such is J. Staples, who looked at the idea of very well-covered graphs, where every maximal independent set must contain exactly half of all the vertices of a graph [33].

M. Lewin is another, who looked at characterizing well-covered line graphs, where the vertices of a line graph L(G) represent the edges of G [27].

A. Finbow and B. Hartnell worked on characterizing well-covered graphs [11], and came up with the following result: If a graph G is of girth eight or greater, G is well-covered if and only if its pendant edges make up a perfect matching.

To demonstrate this idea, Figure 1.9 shows three examples of vertex covers, with the first two graphs being well-covered, and the last, not.

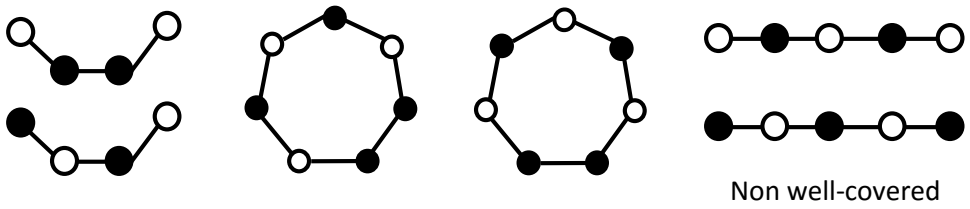

Figure 1.9: Examples of well-covered and not well-covered graphs.

1.3.2 Well Paired-Dominated Graphs

A well-dominated graph, G, is one in which a minimal dominating set of G is always minimum. Mathematician S. Fitzpatrick worked on the concept of well paired-dominated graphs in her doctoral thesis in 1997 [14], and with B. Hartnell in a paper published in 2010 [16]. Their goal was to find a collection of graphs where a minimal paired-dominating set could be found using a greedy algorithm, and such that the minimal set was always minimum.

The approach they use is to place K_2's until all nodes are guarded, calling this set G_0. A K_2, say g, is then removed from the set if G' = G - $\{g\}$ is still a dominating set. This procedure is continued until no more K_2's can be removed, resulting with a minimum paired-dominating set of a fixed size, regardless of the K_2's removed.

As a result, Hartnell and Fitzpatrick found that a connected graph G, with girth at least eight, is a well paired-dominated graph if and only if the stems of G form an independent dominating set in G, or G is a 9 cycle.

Figure 1.10 illustrates the idea of well paired-dominated graphs.

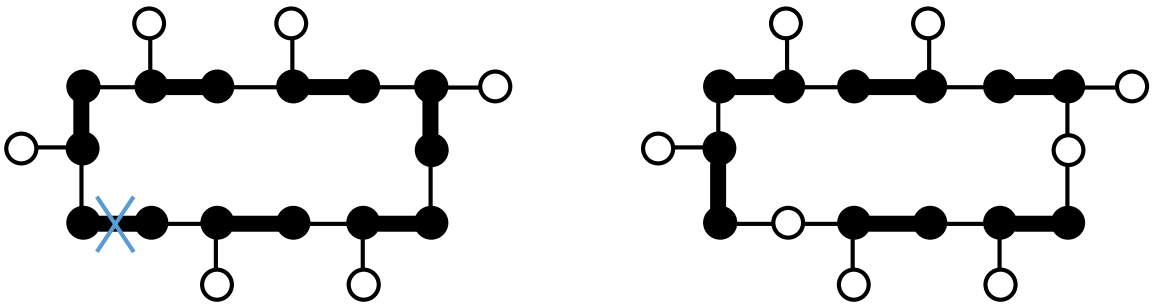

Figure 1.10: Illustrating the idea of well paired-dominated graphs.

1.3.3 Maximum Matchings

A matching in a graph is a set of edges which are all independent from each other. A maximal matching is one such that no more edges can be added to the set, and if an edge is removed, it is no longer maximal. Similarly, a maximum matching is a maximal matching of largest cardinality. An M-alternating path is a path in G, such that the edges are

alternatively in and out of the matching, and an M-augmenting path is an M-alternating path such that neither of the end edges of the path are in M.

Figure 1.11 shows the difference between maximal and maximum matchings.

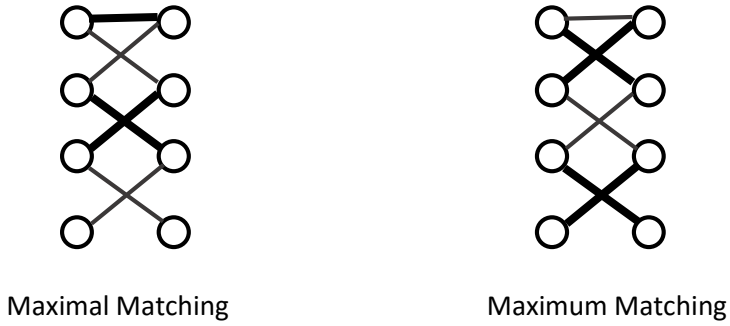

Figure 1.11: Maximal and maximum matchings.

The topic of matchings is very well-known, and is one of the basic concepts to study in graph theory. There are many theorems regarding matchings, and associated problems. One such theorem states that a matching M, in a graph is maximum if and only if there exists no M-augmenting path in G. Another such is Hall's Marriage Theorem, which is based on the idea of matching a set of girls with a set of boys, and looking at when it is possible for each girl to get paired with a boy she knows. The theorem, giving an answer to this, says that a solution exists if and only if every subset of k girls collectively know at least k boys [2]. As it so happens this theorem will be used later on to help with the proof of some bounds for our problem in the next section.

In regards to well-known matching problems, one of the most famous was on finding a maximum matching for a general graph. Edmonds was the first to present a

polynomial algorithm for finding a maximum matching of a graph, in 1965 [10]. We will come to see in Section 4, that Edmond's polynomial algorithm can be used to help characterize graphs for our problem.

2 The Problem

We now move on to the main problem of this work. As mentioned earlier, the problem of integrated domination deals with the idea of combining two different types of domination; standard domination with paired-domination. Here, we look at the idea of guarding a graph using a combination of K_1's and K_2's as guards, with the main problem being to determine a minimum dominating set where one has access to both types of guards. If there are ties for the minimum number of guards, the set with the fewest number of K_2 guards is chosen.

First, we give a bit of background information into why one would want to choose the least number of K_2's, and why it matters. To start, we bring up the notion of the "cost". The cost of a dominating set refers to how much energy, resources, etc., the set requires in order to fully dominate a graph. Perhaps a company wants to hire 10 guards to patrol its building, and so the cost would be the price or salary of the 10 guards, as an example. Generally, the cost, in terms of graphs, would be the number of guards in the dominating set or the domination number of the graph, however this is not the case in this particular problem.

In this problem there are two different types of guards, with each representing a certain style of domination. As a general rule in this problem, it is noted that one K_1 guard

is cheaper, or more cost-efficient, to use than one K_2 guard, which is then cheaper to use than two K_1 guards. The idea behind this is that if one chooses to look at a K_2 as two K_1's, then the amount of resources it would use would be twice that of one K_1. Conversely, if a K_2 is considered as one K_1 moving back and forth between two vertices, then it is still using more energy than a K_1, which stays on one vertex only, but is still cheaper than two K_1's.

A common sense issue arises when one chooses an integrated dominating set of 99 K_2's rather than an integrated dominating set of 100 K_1's, however we chose our rules in order to keep the problem more manageable. Future work could further refine the cost issue.

2.1 Further on the Problem

In spite of the fact that for integrated domination one can use both K_1's and K_2's to dominate a graph, which may be an advantage over the original or paired-domination, there are situations where using solely K_1's or solely K_2's yields an optimal result.

For example, it is best to dominate a P_6 with only K_1's. If one were to use K_2's, or a mix of K_1's and K_2's, overlap would be created and the set would not be minimum. Similarly, a P_8 graph is best dominated using only K_2's, rather than K_1's or a mix of the two. Generally speaking however, most graphs require a combination of K_1's and K_2's to find their minimum integrated dominating set. Figure 2.1 illustrates guarding a P_6 and a P_8 using a variety of guard types.

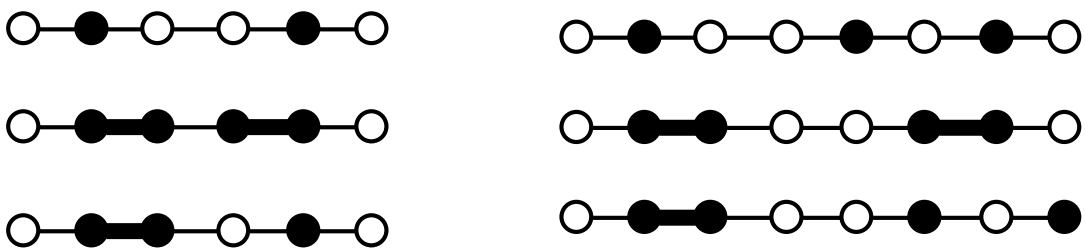

Figure 2.1: Guarding a P_6 and a P_8 with different variations of guards.

Expanding on the idea mentioned above, one of the issues that comes up when looking at this problem is that it is possible to have multiple minimal integrated dominating sets with the same cardinality, yet with different numbers of K_1's and K_2's. Figure 2.2 illustrates this issue.

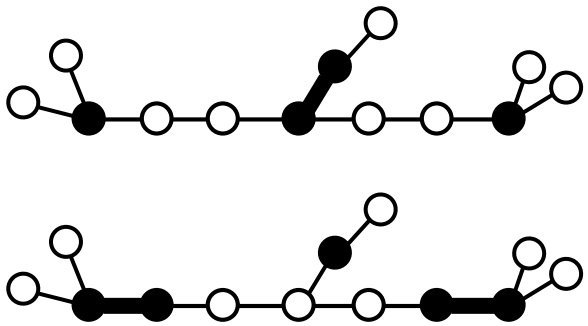

Figure 2.2: Two different minimal integrated dominating sets with the same cardinality.

For the graph in Figure 2.2, if we were to start at either the right or left branches, possibly by starting at the end of a longest path, a minimal integrated dominating set could result with two K_1's and only one K_2. However, if one were to start on the shortest branch, they could end up with one K_1 and two K_2's. Both sets contain three guards whose closed neighbourhoods partition the graph, however clearly the option with only one K_2 is the

minimum integrated dominating set. Thus, this not only is a problem of knowing how many guards to have in the integrated dominating set, but also of knowing which guard to place where.

2.2 The Goal of This Work

Finding an algorithm for obtaining a minimum integrated dominating set appears to be quite difficult. The goal of this work is to give some insight into the problem, and to give a collection of graphs where a minimum integrated dominating set can efficiently be found. This work also gives a set of subgraphs, which if found in a graph, one knows exactly how to guard them; and gives a bound on the cardinality of a minimum integrated dominating set, which we look at next.

2.3 Bounds

Theorem 2.1 In a graph with no isolates, the cardinality of a minimum integrated dominating is equal to the cardinality of a minimum paired-dominating set.

Proof: Assume a minimum integrated dominating set say D, consists of $R_1 \cup R_2$ where R_1 is a set of K_1's and R_2 is a set of K_2's. If $R_1 = 0$, then D is a paired-dominating set and we are done.

If $R_1 \neq 0$, let T be all the neighbours of vertices in R_1 that are not in the integrated dominating set D. Note that each vertex in R_1 has such a neighbour, as if no such neighbour existed for a vertex v in R_1, then v would not need to be in D.

If any subset, say R, of R_1 had the property that $|N(R)| < |R|$, then $(D \backslash R) \cup (N(R))$ would be a smaller integrated dominating set than D, which contradicts D being a minimum integrated dominating set. Now, if every subset R of R_1 is such that $|N(R)| \geq |R|$, then Hall's theorem guarantees that there is a matching of vertices in T that saturate the vertices in R_1. Although Hall's theorem considers bipartite graphs, our situation here can be represented as a bipartite graph when we consider the vertices of R_1 with their neighbours in T.

Now if we use K_2's on this matching, we have a minimum paired-dominating set of the same cardinality as D and hence the proof is complete. ■

3 Subgraphs

To begin, we look at a variety of subgraphs which can efficiently be guarded minimally, regardless of their location in a graph. In some of these examples, the concept of removing a vertex from a graph is discussed, and simply refers to the fact that if a vertex is guarded, with no ‘more efficient’ solution existing, then it can be removed from the graph. This is to make the remainder of the graph smaller and hopefully easier to work with.

Note: As a reminder, in a graph, a stem is a vertex which has at least one neighbour of degree one. This particular neighbour is called a leaf, or an endvertex, and is characterized by having only one edge incident with it, or in other words, having only one neighbour.

Lemma 3.1 In a graph G, there is a minimum integrated dominating set D such that every stem hosts a guard from D.

Proof: In order to dominate a graph G, every vertex in G must be dominated by at least one vertex in the integrated dominating set D. Let u be an endvertex, and v be the stem

adjacent to u. In order to dominate u, either u or v must be in D (See Figure 3.1). If u is in D as a K_1, it will dominate only itself plus v. If v is in D as a K_1, it will dominate itself, u, and all of its other neighbours. Thus, one does at least as well, if not better, to have a guard on a stem. We note that such a guard may be a K_1 or part of a K_2. ■

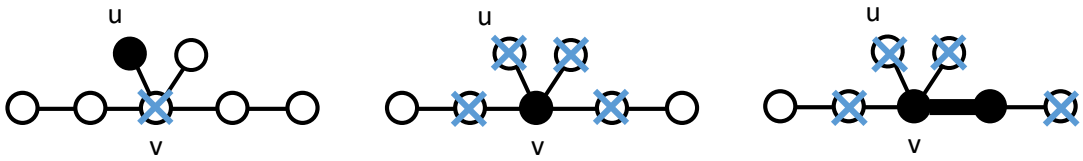

Figure 3.1: Options for guarding a stem.

Lemma 3.2 Let G be a graph with a minimum integrated dominating set D. If there exists a stem x, whose only non-leaf neighbour, say y, is of degree 2, and y's only other neighbour is a stem, z, then x will host a K_1 guard.

Proof: As both x and z are stems, they will both require a guard. There are two options for possible integrated dominating sets; either a K_1 is placed on x, or a K_2 is placed between x and y (See Figure 3.2). The only advantage to having a K_2 between x and y would be that it guards z, however since z is a stem it requires its own guard, to guard its leaves. Therefore, the K_2 does no more guarding than a K_1 in this spot, and so a K_1 is the cheaper option and a minimum solution. ■

Corollary 3.2 A minimum integrated dominating set for a graph G described in Lemma 3.2 will be a minimum integrated dominating set for G\N[x], along with the vertex x hosting a K_1.

Proof: After removing x and its leaves, vertex y then becomes a leaf of z, and since it is being dominated by z, it can safely be removed from the graph.

Figure 3.2: Options for guarding the subgraphs described in Lemma 3.2.

Lemma 3.3 If adjacent to the end of a longest path in a tree T, there exists a stem x, whose only non-leaf neighbour y, is not a stem and is of degree ≥ 3, then any minimum integrated dominating set can be transformed in polynomial time to one in which x hosts a K_1 guard.

Proof: As y is near the end of a longest path, is of degree ≥ 3 and is not a stem, it must have a stem neighbour other than x, say z.

To dominate the graph, a K_1 could be placed on x, or a K_2 between x and y (See Figure 3.3). The advantage to placing a K_2 between x and y is that it would not only guard y, but y's neighbours as well. However, since z, a neighbour of y, is a stem, it will need its own guard, which would also guard y.

In a case where another neighbour of y needs to be guarded, one can place a K_2 between z and y to cover that neighbour. Thus, one would do no better to place a K_2 between x and y, and so a K_1 on x gives an optimal solution. ■

Corollary 3.3 A minimum integrated dominating set for a graph G described in Lemma 3.3 will be a minimum integrated dominating set for the graph of G without x and its leaves, along with vertex x hosting a K_1.

Proof: Vertex x and its leaves can be removed from G, as x will only host a K_1. Since y is of degree ≥ 3 and does not become a leaf of z, it's possible that y may need to host a guard later on in order to guard its neighbours, and so only x and its neighbours are removed. ■

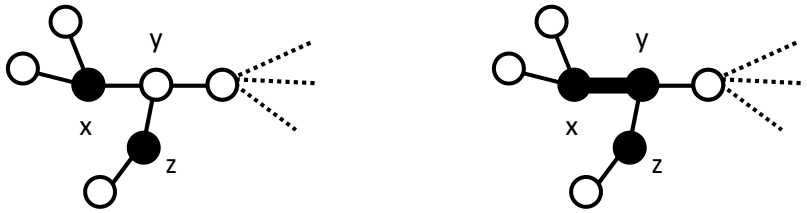

Figure 3.3: Options for guarding the subgraphs described in Lemma 3.3.

Lemma 3.4 If adjacent to the end of a longest path in a tree T, there is a stem x, whose only non-leaf neighbour is a vertex, say y, of degree 2, where y's other neighbour z, is not a stem and is of degree ≥ 3, then a minimum integrated dominating set can be transformed in polynomial time where x hosts a K_1 guard.

Proof: To dominate the graph, a K_1 could be placed on x or a K_2 could be placed between x and y. The advantage to placing a K_2 between x and y is that it would guard z. However, since z is near the end of a longest path, is not a stem and is of degree ≥ 3, z must contain at least one other branch, of length two or three.

If the branch is of length 2, it will require a guard on its stem, which is adjacent to z, and so z would be guarded. If the branch is of length 3, it will require a guard on its stem, which can easily change to a K_2, and so will again guard z.

Therefore, to have a K_2 between x and y in order to guard z would be unnecessary, as z is just as easily dominated by one of its adjacent vertices on its other branch, and so having x host a K_1 guard is an optimal solution (See Figure 3.4). ■

Corollary 3.4 A minimum integrated dominating set for a graph G described in Lemma 3.4 will be a minimum integrated dominating set for G\N[x], along with the vertex x hosting a K_1.

Proof: Vertex x and its leaves can be removed from the graph, and since y would then become a leaf of z and would be guarded by x, it can also be removed. ■

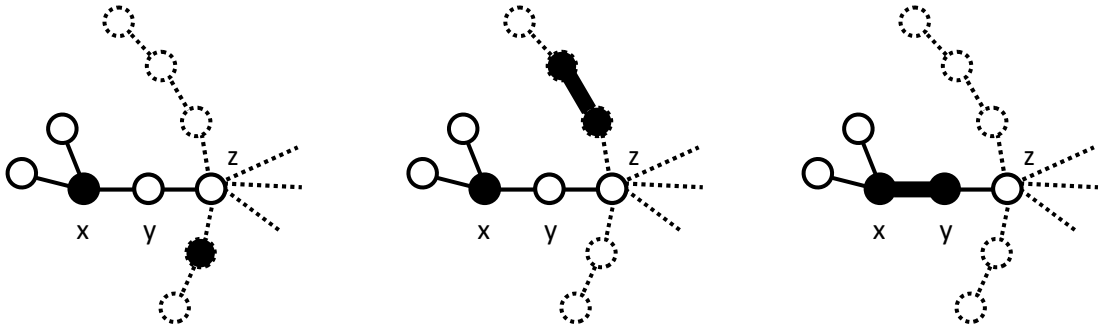

Figure 3.4: Options for guarding the subgraphs described in Lemma 3.4.

Lemma 3.5 Let G be a graph with a stem x, whose only other neighbour besides its leaves is another stem y, and y is adjacent to only leaves and one other vertex, say z. Then if D is

a minimum integrated dominating set, then D can be transformed in polynomial time to a dominating set in which x and y will host a K_2 guard between them.

Proof: Either z is in the minimum integrated dominating set D, or z is in V\D. As both x and y are stems, they will both require a guard. If z hosts a guard in D, then a K_1 could be placed on x and a K_2 between y and z, or a K_2 could be placed between x and y, and z would have either a K_1 or a K_2 between itself and another node if needed. Thus, a K_1 on x and a K_2 between y and z would do no better than a K_2 between x and y.

If z is not in D, then x and y will both still require guards, and a K_2 between these nodes would be more efficient than two K_1's. Therefore, an optimal solution will have a K_2 between x and y. ■

(See Figure 3.5).

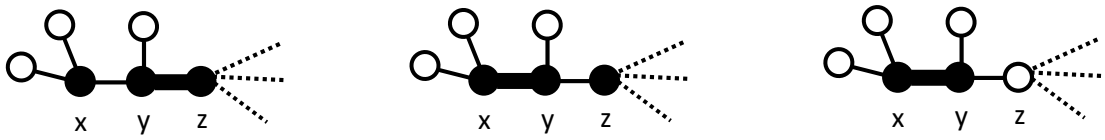

Figure 3.5: Options for guarding the subgraphs described in Lemma 3.5.

Lemma 3.6 Let T be a tree with a stem x, adjacent to the end of a longest path in T. Assume that x's only non-leaf neighbour is a vertex y, of degree 2, where y has only z, a degree 2 vertex, as its other neighbour, and z's only other neighbour w, is not a stem and is of degree ≥ 3. Further assume that vertex w is adjacent to one stem, say v, and v is only adjacent to one non-leaf vertex. If D is a minimum integrated dominating set, then D can be

transformed in polynomial time to a dominating set in which x hosts a K_1, and v and w host a K_2 between them.

Proof: Vertices x and v are both stems, and so both require a guard. If a K_1 is placed on x, it guards x's leaves, x and y. This means z is still unguarded, and so a K_2 should be placed between v and w in order to guard z, which in turn guards all of w's other neighbours. If a K_1 is placed on v instead, it only guards v's leaves plus w, and z becomes stuck between two already guarded vertices.

If a K_2 is placed between x and y, it will guard x's leaves, x, y and z, leaving it so that a K_1 can be placed on v, in order to guard w. However, here w's neighbours aren't yet guarded. Alternatively, a K_2 could be placed between v and w to guard w's other neighbours, however this would use more K_2's and does no better than having a K_1 on x (See Figure 3.6).

Therefore, there is an optimal solution with x hosting a K_1 and v and w hosting a K_2. ■

Corollary 3.6 A minimum integrated dominating set for a graph G described in Lemma 3.6 will be a minimum integrated dominating set for G\N[x], along with the vertex x hosting a K_1.

Proof: Vertex x, its leaves and y can be removed from the graph as they are all guarded and are only adjacent to vertices that are already guarded. ■

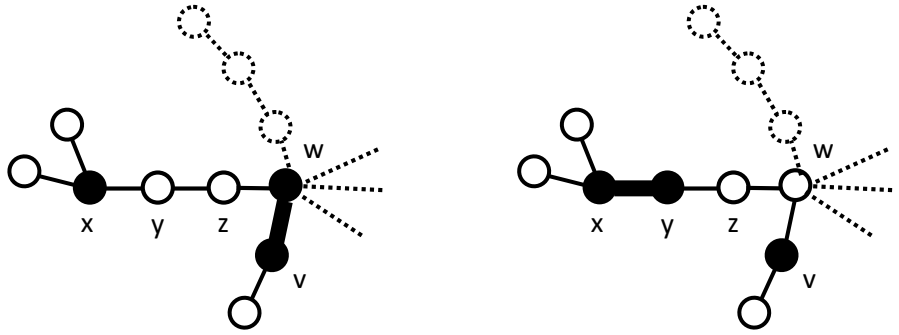

Figure 3.6: Options for guarding the subgraphs described in Lemma 3.6.

Lemma 3.7 If in a graph G there exists a vertex x, which is part of an induced K_n, such that the removal of x disconnects the remaining vertices of the K_n from the rest of G, then a minimum integrated dominating set D can be found with x hosting a guard in D.

Proof: Let K be the set of vertices in the K_n subgraph. Let x be the vertex in K such that x is the only vertex connecting G[K] with the remainder of G.

Let S denote the other neighbours of x, not in K. If x is in the minimum integrated dominating set D as a K_1, then x guards itself plus all of S and K, since all vertices in a complete graph are adjacent. On the other hand, if a vertex v, in K\{x}, is in D as a K_1, it will guard only x and the remaining vertices in K, with none of the vertices in S being guarded. Thus, there is a minimum integrated dominating set containing x. ■

(See Figure 3.7).

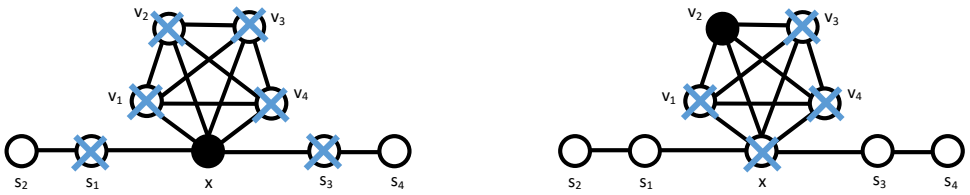

Figure 3.7: Options for guarding a clique.

Lemma 3.8 In a graph G, suppose there exists a stem x, whose non-leaf neighbours are all of degree 2. If these neighbours are also adjacent to vertices of degree 2 and their other neighbours are all stems, then in a minimum integrated dominating set x will host a K_1.

Proof: Let $\{a_1, a_2, \ldots, a_i\}$ be the non-leaf neighbours of x. Let $\{b_1, b_2, \ldots, b_i\}$ be the other neighbours of $\{a_1, a_2, \ldots, a_i\}$. Let $\{s_1, s_2, \ldots, s_i\}$ be the remaining neighbours, and thus stems, adjacent to $\{b_1, b_2, \ldots, b_i\}$.

As x is a stem, it will require a guard, which can be either a K_1 or a K_2. If it is a K_1 then it will guard x, its leaves, plus all $\{a_1, a_2, \ldots, a_i\}$. As $\{s_1, s_2, \ldots, s_i\}$ are stems, they will each require a guard, which will guard themselves and all their neighbours, including each of the adjacent $\{b_1, b_2, \ldots, b_i\}$.

If x is in as a K_2, it will go between x and one of the $\{a_1, a_2, \ldots, a_i\}$, say a_n. Let a_n be adjacent to b_n from $\{b_1, b_2, \ldots, b_i\}$. Then the K_2 will guard x, x's leaves, all of the $\{a_1, a_2, \ldots, a_i\}$ plus b_n. However, b_n is adjacent to a stem, which will have a guard, and so having a guard between x and a_n is not necessary. Therefore, a K_1 on x is more efficient, and so gives an optimal solution. ■

(See Figure 3.8).

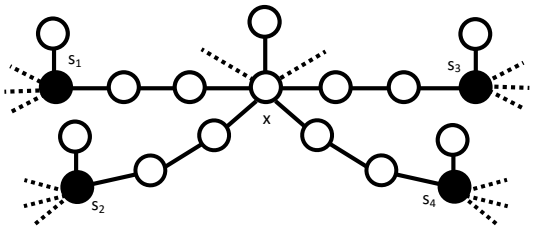

Figure 3.8: Guarding an isolated stem.

Lemma 3.9 In a graph G, suppose there exist two adjacent stems x and y, whose other non-leaf neighbours are all of degree 2. If these neighbours are also adjacent to vertices of degree 2 and their other neighbours are all stems, then in a minimum integrated dominating set x and y will host a K_2 between themselves.

Proof: As in Lemma 3.8, it is shown that to put a K_2 between either x or y, and one of their adjacent neighbours, would be redundant. Thus, two K_1's could be placed on x and y, however a K_2 between these vertices is more efficient, and therefore is found in a minimum integrated dominating set for this type of graph. ■

(See Figure 3.9).

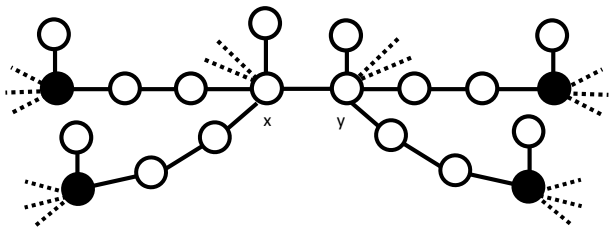

Figure 3.9: Guarding a pair of isolated stems.

Lemma 3.10 If in a graph G, there exist two adjacent non-stem vertices x and y, with their only other neighbours being stems, then the vertices x and y can be removed from the graph.

Proof: Since each of the neighbours of x and y are stems they will each require a guard, and so x and y will be guarded. As x and y are not stems, and have no other neighbours besides themselves and stems, they need not be in the integrated dominating set, as having them in the integrated dominating set gives no advantage. ■

Thus, the removal of the vertices x and y does no harm to the dominating set, and potentially breaks the graph into two smaller graphs. This would hopefully make the graphs easier to handle (See Figure 3.10).

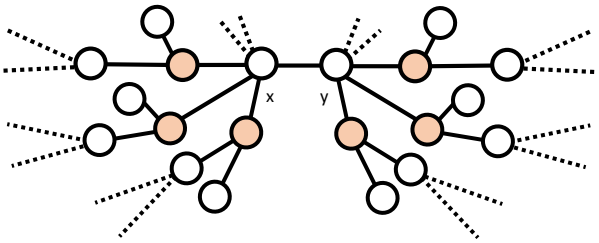

Figure 3.10: Guarding a stem covered edge.

Note: Here we will talk about how many guards are needed for certain path lengths. Lemma 4.1 in Section 4 explains how to find the number of K_1's and K_2's to guard different path lengths, and goes into further detail on this topic.

The basis of these results is formed from the idea that a path of length 5 cannot be guarded efficiently using any combination of K_1's and K_2's. One K_1 is not sufficient to completely dominate a P5, nor is one K_2. Two K_1's create overlap, and the same is true for

two K_2's, or one K_1 plus one K_2. Thus, part of the strategy for these paths was to avoid "creating" paths of length 5 when not necessary.

It is also noted that using as many K_2's as possible to guard long paths is more efficient than using K_1's, as K_2's can guard four vertices on a path in total, whereas K_1's can only guard three.

Therefore, the aim was to use as many K_2's as possible, without forcing w into a situation where it would be left with a path of 5 unguarded nodes.

Lemma 3.11 If in a graph G, there exists a vertex w, such that $\deg(w) \geq 3$, and w has a k-arm with $k \geq 7$, then the following statements hold for a minimum integrated dominating set:

a) For k being a 4t-arm, with t a variable, K_2's should be placed starting at the tail-end of the arm, and working in towards w, stopping when there are 8 unguarded nodes left between the K_2's and w.

b) For k being a 4t+1-arm, K_2's should be placed starting at the tail end of the arm and stopping when there are 9 unguarded nodes left between the K_2's and w.

c) For k being a 4t+2-arm, K_2's should be placed starting at the tail end of the arm, and stopping when there are 10 unguarded nodes left between the K_2's and w.

d) For k being a 4t+3-arm, K_2's should be placed starting at the tail end of the arm, and stopping when there are 7 unguarded nodes left between the K_2's and w.

Proof: Here it is assumed that the rest of G, without w and the k-arm, has not yet been guarded, and so the status of w is unknown.

In each of the four cases in Lemma 3.11, in an optimal arrangement of guards for the entire graph G, w can be in one of three positions (See Figure 3.11):

Case 1: Vertex w can host a guard, in which case w plus one neighbour from the adjacent arm are guarded.

Case 2: Vertex w is adjacent to a guard not on the k-arm and so is guarded.

Case 3: Vertex w is two or more vertices away from any guard not on the k-arm and so still needs to be guarded.

Using this, we now look at each of the possible lengths separately.

a) Let G be a graph with a 4t-arm adjacent to a vertex w, of degree ≥ 3. If 8 nodes are left unguarded, the guarded section of the arm becomes of length 4t – 8, and thus would use t-2 K_2's.

In Case 1, where w hosts a guard and so one vertex of the arm is guarded, the unguarded length of the arm becomes 8 – 1 = 7, which can be guarded using one K_1 and one K_2. In Case 2, w is guarded and so the length of the unguarded section stays 8, and can be guarded using two K_2's. For the final Case 3, the length of the unguarded section becomes 8 + 1 = 9, as w will need to be guarded, and so can easily be guarded using three K_1's.

Adding an extra K_2 to the arm, and so leaving 4 vertices unguarded, would create overlap. In the first case the unguarded length becomes 3, and can be guarded with one K_1.

In Case 2, the length stays at 4 and can be guarded with one K_2. However, in the third case the unguarded length becomes 5, and as we have shown, can not be efficiently guarded and creates overlap. Therefore, leaving 8 vertices unguarded after placing the K_2's, is an optimal solution (See Example A).

b) Similarly, for a 4t+1-arm, leaving 9 vertices unguarded before w results in a guarded section of length 4t+1 – 9, using t-2 K_2's. For Case 1, the unguarded length becomes 9 - 1 = 8, guarded with two K_2's, Case 2 leaves the unguarded length at 9, which can be guarded by three K_1's, and Case 3 leaves 9 + 1 = 10 vertices unguarded, which can use one K_2 and two K_1's.

Adding an extra K_2 at the start would leave 5 vertices unguarded, which would create overlap in Case 2.

c) For 4t+2-arms, leaving 10 vertices unguarded between the K_2's and w would result in the length of the guarded section being 4t+2 – 10 and would use t-2 K_2's. Using the same procedure as before, Case 1 would see 9 vertices unguarded, Case 2 would have 10 vertices unguarded, and Case 3 would have 11 vertices unguarded.

Again, adding an extra K_2 at the start would leave 6 vertices unguarded, which would create overlap in the first case.

d) Finally, for 4t+3-arms, placing K_2's up until there are 7 nodes unguarded before w forces 4t+3 – 7 vertices, guarded by t-1 K_2's. For Case 1, the unguarded length would be 6, Case 2 would give an unguarded length of 7, and Case 3 would leave 8 vertices unguarded.

Adding an extra K_2 at the start would leave 3 vertices unguarded, and would create overlap in Case 1. ■

Table 3.1 gives the lengths of the unguarded sections for each of the 4t, 4t+1, 4t+2 and 4t+3-arms in each of the three cases, and shows where the overlap would be in each situation.

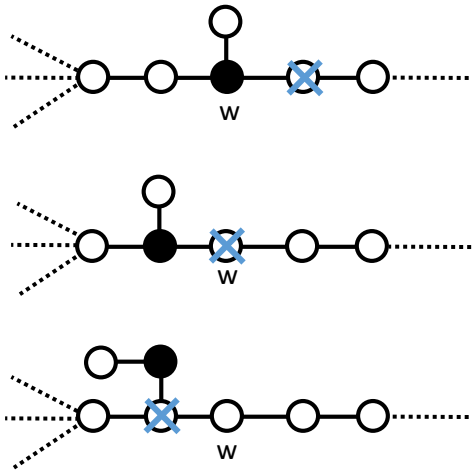

Figure 3.11: Positions of a high degree vertex.

Table 3.1: Case results for each length of k-arms.

	Unguarded Length			**Where the overlap is**
4t	**Case 1**	**Case 2**	**Case 3**	
Leaving 8 unguarded (*before w*) (Using t-2 K_2's)	7	8	9	
Leaving 4 unguarded (Using t-1 K_2's)	3	4	5	Case 3
4t+1				
Leaving 9 unguarded (Using t-2 K_2's)	8	9	10	
Leaving 5 unguarded (Using t-1 K_2's)	4	5	6	Case 2
4t+2				
Leaving 10 unguarded (Using t-2 K_2's)	9	10	11	
Leaving 6 unguarded (Using t-1 K_2's)	5	6	7	Case 1
4t+3				
Leaving 7 unguarded (Using t-2 K_2's)	6	7	8	
Leaving 3 unguarded (Using t-1 K_2's)	2	3	4	Case 1

Example A: Figure 3.12 illustrates a 12-arm being guarded for each of the three cases, with the first part showing what would happen when 8 vertices are left unguarded, and the second part showing when 4 vertices are left unguarded.

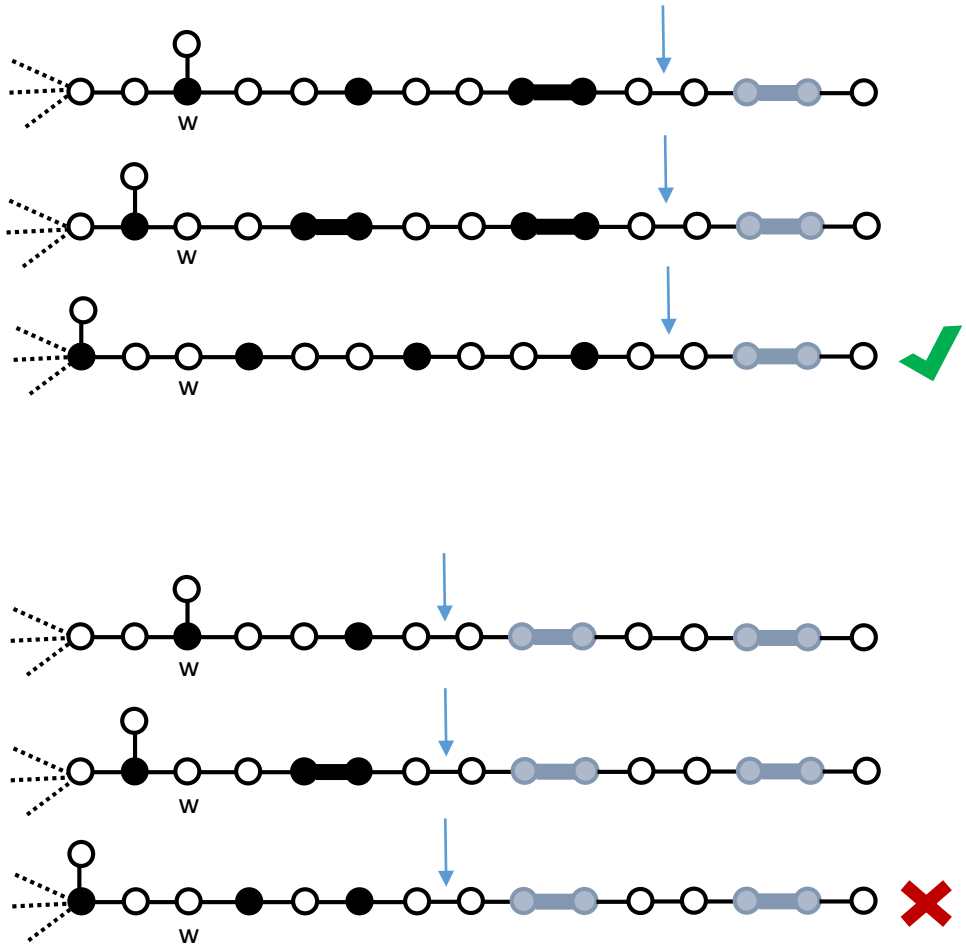

Figure 3.12: Guarding a 12-arm where 8 and 4 vertices are initially left unguarded.

4 Graphs

We now move on to specific graphs where a minimum integrated dominating set can efficiently be found.

Note: Two types of graphs that can be guarded easily using K_1's and K_2's are paths, P_n, and cycles, C_n. Paths, as mentioned earlier, are strings of adjacent degree 2 vertices, with only the two end vertices being of degree 1, and each vertex in the path being distinct. A path with 6 vertices would be denoted as P_6. Cycles have a similar characteristic, but are without end vertices, and every vertex is of degree 2. A cycle with 6 vertices would be called a 6-cycle, or a C_6.

Lemma 4.1 For paths and cycles a minimum integrated dominating set can be found efficiently.

Proof: As each K_1 can guard 3 vertices, and each K_2 can guard 4 vertices, the length of the path or cycle can be divided into sections of multiples of 3 and 4, and a counting formula

can be used to find the number of guards needed and of which type. The formulas are shown in Table 4.1, with t as a variable.

Table 4.1: Path length formulas.

Length	Number of K_2's	Number of K_1's
4(t) + 0	t	0
4(t) + 1	t-2	3
4(t) + 2	t-1	2
4(t) + 3	t	1

Let $R = \{r_1, r_2, \ldots, r_n\}$ be the guards found using the appropriate formula above. For paths, one starts at a leaf and places one of the guards, say r_1, on the node adjacent to the leaf. For every vertex r_1 dominates, that vertex is removed from the graph. Then, the process is repeated, using a guard from $R\backslash\{r_1, \ldots, r_k\}$ where $\{r_1,\ldots, r_k\}$ are the guards already chosen.

For cycles, the same procedure is followed, however as there are no stems, and each vertex in C_n has the same properties, one starts by choosing an arbitrary vertex to place the first guard.

Of course for the smaller path or cycle lengths, 1, 2 and 5, the formulas don't hold. In the cases where the path or cycle length is 1 or 2 nodes, a single K_1 will suffice. When the length is 5, overlap is unavoidable, and so using two K_1's is optimal. ■

Figure 4.1 illustrates guarding different path lengths according to the formulas given above.

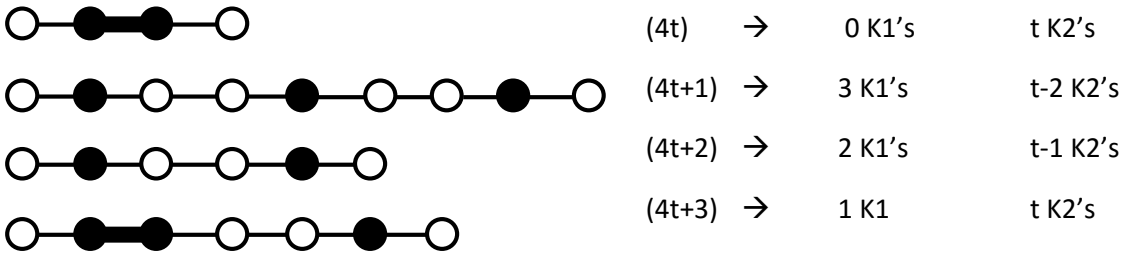

Figure 4.1: Guarding paths of different lengths.

Lemma 4.2 If G is a graph such that its stems form an independent dominating set, then a minimum integrated dominating set can efficiently be found.

Proof: Let S be the set of stems in G. Since S forms a dominating set, the stems must, as a whole, be adjacent to every vertex in G\S, and so having a guard on each stem will dominate every vertex in G. Since S is an independent set, no two stems are adjacent, and so there is no advantage to using K_2's. Therefore, an optimal solution can be obtained by placing a K_1 on each stem in G. ■

Notice that this characterization also applies to well paired-dominated graphs, as we discussed in Section 1.

Corollary 4.2 If in a graph G, the closed neighbourhoods of every stem form a partition of V(G), then a minimum integrated dominating set can efficiently be found with a K_1 on every stem.

Proof: Let G be a graph with the closed neighbourhoods of all stems in G partitioning the graph. Let s_1 be an arbitrary stem in G with neighbours $\{v_1, v_2, \ldots, v_m\}$. For each v_i, its only neighbours other than s_1, are vertices that are adjacent to other stems, say $\{r_1, r_2, \ldots, r_m\}$. If a K_1 is placed on s_1, then $\{v_1, v_2, \ldots, v_m\}$ are guarded. A K_1 on each of $\{r_1, r_2, \ldots, r_m\}$ will then guard all the other neighbours of $\{v_1, v_2, \ldots, v_m\}$.

The only advantage to placing a K_2 between s_1 and one of its neighbours, say v_1, is that v_1's neighbours will then also be guarded. However, since v_1's neighbours are all adjacent to stems, they will be guarded by their adjacent stem, and thus it does no better to place a K_2 on s_1 rather than a K_1. ■

(See Figure 4.2).

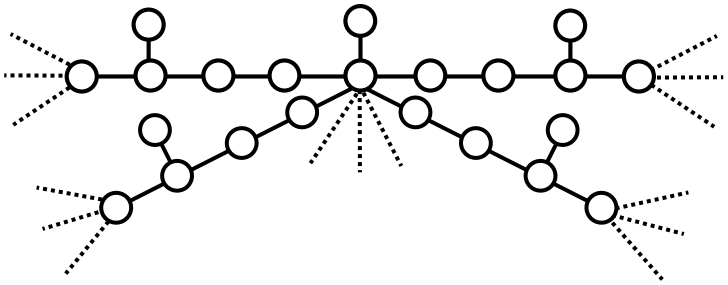

Figure 4.2: Example of a partition of stems

Lemma 4.3 If in a graph G, all stems are found in adjacent pairs, the closed neighbourhoods of each adjacent pair of stems is disjoint from all others, and the closed neighbourhoods form a partition of V(G), then every pair of stems will host a K_2 between them in a minimum integrated dominating set.

Proof: From the results of Corollary 4.2, it is shown that in these types of graphs where the closed neighbourhoods of the stems form a partition of the graph, to place a guard between one of the stems and a non-stem neighbour is redundant. Further, when two stems are adjacent and their neighbourhoods form a partition, the two optimal solutions are to either place two K_1's on the stems, or a K_2 between them. However, as shown before, one K_2 is a more optimal solution than two K_1's, and thus a K_2 is placed between all adjacent pairs of stems. ■

Lemma 4.4 If in a graph G, the subgraphs induced by the set of stems are K_1's and K_2's, and the closed neighbourhoods of these induced subgraphs are disjoint from all other closed neighbourhoods of the induced subgraphs, and these neighbourhoods form a partition of V(G), then a minimum integrated dominating set has a K_1 on each K_1 subgraph of stems, and a K_2 on each K_2 subgraph of stems.

Proof: Combining the results from Corollary 4.2 and Lemma 4.3, it's shown that this is a optimal solution. ■

Lemma 4.5 If G is a graph in which each vertex is either a stem or a leaf, then a minimum integrated dominating set can efficiently be found.

Proof: As each vertex is either a stem or a leaf, to place a guard on each stem will dominate every vertex in the graph. Wanting to use the fewest number of guards possible, we want to pair up as many stems as possible to use K_2's. This now, in a sense, becomes a problem

of finding a maximum matching. As mentioned in Section 1, a polynomial time algorithm exists to solve this problem, due to J. Edmonds' results [10], and so we can use this algorithm to find where the K_2's should be placed, and then place K_1's on the remaining stems. ■

The next few graphs we characterize are ones where each high degree vertex is a stem. These graphs therefore consist only of stems, leaves and k-paths. For these k-paths, we note that each stem can be guarded according to its length.

As each stem requires a guard, a K_1 is placed on all stems in the graph as a placeholder. These K_1's may be changed later to K_2's. Now, between each set of stems, there is a path that needs to be guarded. For each of these paths, there are three possibilities for guarding it, shown in Table 4.2.

Table 4.2: Options for boundary stems.

Option	**Action**
Option 1	Both of the K_1's on the two boundary stems are left as K_1's.
Option 2	One of the K_1's on the boundary stems is changed to a K_2, going between the stem and an adjacent vertex on the path.
Option 3	Both of the K_1's on the boundary stems are changed to K_2's, going between the stems and adjacent vertices on the path

In the first case, each of the K_1's would guard one vertex on the path, leaving an unguarded section of two vertices less than the original length. In the second case, the K_1 would guard one vertex on the path, and the change to a K_2 for the other stem would guard two vertices on the path, leaving an unguarded section of three vertices less than the original length. For the third case, the change to K_2's for both stems leaves an unguarded section of four vertices less than the original length.

For smaller k-paths, of lengths 1, 2, 3, 4 and 5, slight exceptions are made, and these results are shown in Table 4.3, which shows whether the boundary stems should have K_1's, K_2's, or both. Note that for a path of length 5, an extra K_1 is required on the path as well.

Table 4.3: Optimal options for shorter k-paths.

Length of the Path	Optimal Option for Boundary Stems
1	2 K_1's
2	2 K_1's
3	1 K_1 & 1 K_2
4	2 K_2's
5	2 K_1's

In the more general cases however, we have general guidelines as to how to guard each length. First, note guarding as many vertices as possible using K_2's is best, as this guards the most vertices using the fewest number of guards, and so paths of length 4t are

considered optimal here as they can be guarded in this way. This is similar to the idea in Lemma 3.11.

For the k-paths of length 4t, choosing Option 3 and changing both of the boundary K_1's to K_2's still leaves the unguarded path at a length of 4t, and so is optimal. Here, one would use t-1 K_2's for the path, as its length would be 4t – 4, plus 2 K_2's for the boundary stems, using t+1 K_2's total.

If one were to choose Option 1, the unguarded length would become 4t – 2, and so would use t-2 K_2's and 2 K_1's for the length, plus 2 K_1's as a boundary guards, giving a total of t+2 guards. Similarly for choosing Option 2, the unguarded length becomes 4t – 3, and would use t-3 K_2's and 3 K_1's, plus 1 K_1 and 1 K_2 as boundary guards, and thus use a total of t+2 guards.

Clearly, as each option will use two boundary guards regardless, forcing an unguarded length of 4t is optimal. Figure 4.3 shows the different options for guarding a k-path of length 4t. For the other three lengths, Table 4.4 gives the results for each option.

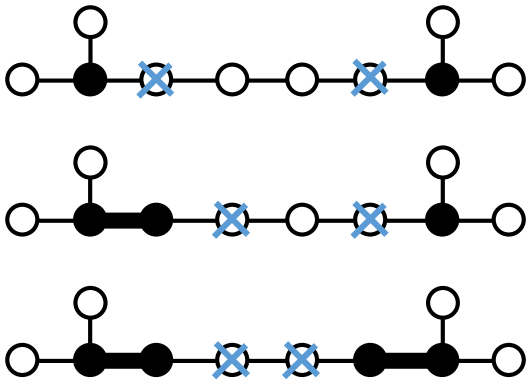

Figure 4.3: Options for guarding a k-path of length 4t.

Table 4.4: Case results for each k-path length.

	Option 1: two K_1's	Option 2: one K_1, one K_2	Option 3: two K_2's
4t	Unguarded Length: 4t-2 Guards Needed: t-2 K_2's 4 K_1's	Unguarded Length: 4t-3 Guards Needed: t-2 K_2's 4 K_1's	Unguarded length: 4t-4 Guards Needed: t+1 K_2's 0 K_1's
4t+1	Unguarded Length: 4t-1 Guards Needed: t-1 K_2's 3 K_1's	Unguarded Length: 4t-2 Guards Needed: t-1 K_2's 3 K_1's	Unguarded Length: 4t-3 Guards Needed: t-1 K_2's 3 K_1's
4t+2	Unguarded Length: 4t Guards Needed: t K_2's 2 K_1's	Unguarded Length: 4t-1 Guards Needed: t K_2's 2 K_1's	Unguarded Length: 4t-2 Guards Needed: t K_2's 2 K_1's
4t+3	Unguarded Length: 4t+1 Guards Needed: t-2 K_2's 5 K_1's	Unguarded Length: 4t Guards Needed: t+1 K_2's 1 K_1's	Unguarded Length: 4t-1 Guards Needed: t+1 K_2's 1 K_1's

Notice that for lengths 4t and 4t+3, there is an option which results in fewer guards, or a cheaper set of guards needed, and in both cases it is required to change at least one of the boundary K_1's to a K_2. With this, a problem arises when two k-paths of lengths 4t or 4t+3 are adjacent to each other, as both require a stem to be changed, however there is only one stem between the two paths. Figure 4.4 illustrates the issue when multiple 4t+3 and 4t paths are adjacent.

Following on this idea, for the cases with lengths 4t+1 and 4t+2, the number and type of guards needed are the same for each of the three options. However, if a path of length 4t+1 or 4t+2 is adjacent to one of length 4t or 4t+3, then clearly choosing an option that doesn't require the changing of a boundary guard is best, as it allows the 4t or 4t+3 path to use that guard instead. Table 4.5 gives the optimal guarding solutions for each k-path length.

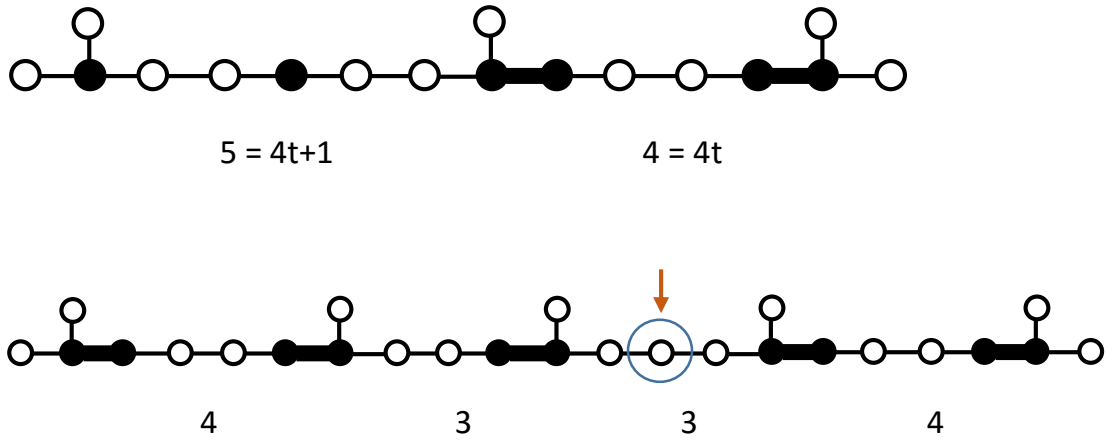

Figure 4.4: Illustrating the issue with multiple k-paths of length 4t.

Table 4.5: Optimal guarding solutions for each k-path length.

Length	Best Option
4t	Change BOTH -both K_1's are changed to K_2's
4t+1	Change NONE -K_1's are kept as K_1's
4t+2	Change NONE -K_1's are kept as K_1's
4t+3	Change ONE -one K_1 is changed to a K_2

This being said, we now move on to some characterizations of graphs of this type.

Lemma 4.6 If G is a graph such that each high degree vertex is a stem, and the only k-paths are of lengths 4t+1 and 4t+2, then a minimum integrated dominating set can be found in polynomial time.

Proof: If our graph consists of only k-paths of lengths 4t+1 and 4t+2, then we can simply place K_1's on all the boundary stems, and the remaining unguarded path lengths can be guarded based on their lengths. ■

Lemma 4.7 For trees with the restriction that each high degree vertex is a stem and every k-path is of length 4t+1, 4t+2 or 4t+3, a minimum integrated dominating set can efficiently be found.

Proof: Let T be a tree satisfying the conditions of the lemma. Each stem requires a guard, so to start, K_1's are placed on all stems in the graph. As shown before, k-paths of length 4t+3 are optimally dominated when one of their boundary K_1's are changed to a K_2.

If no 4t+3 k-path exists in T, then only 4t+1 and 4t+2 k-paths exist, and so each k-path can be guarded in whichever order, using Option 1 (see Table 4.2), and the corresponding path length formula.

Otherwise, there exists at least one k-path of length 4t+3, located at the end of a branch, or adjacent to a string of consecutive 4t+1 or 4t+2 paths which reach the outer end

of a branch. Like this, the 4t+3 k-path has a stem that is on the outside of the tree (adjacent to only leaves), or adjacent to a k-path that doesn't need its boundary stems changed.

Starting at the outside end of a k-path, the tree is guarded by moving up the path towards the centre of the tree, guarding the 4t+1 and 4t+2 k-paths according to Table 4.5 and the corresponding path length formulas, and guarding the 4t+3 k-paths by changing the first, outer, boundary stem to a K_2, and then guarding the rest of the path according to its length. The 4t+3 k-path's other boundary stem is temporarily left as a K_1 in the case that it is part of an adjacent 4t+3 k-path, and thus may need to change to a K_2. This is done until a vertex with ≥ 2 branches attached is reached. When such a vertex is reached, one stops and restarts the procedure at the end of a different branch of T. When no new branches remain, the guarded vertices, minus the vertices with ≥ 2 branches and their neighbours, can be removed from T. This creates new branches for the tree, and the procedure is continued. When only one k-path remains, it can be guarded according to Table 4.5 and the corresponding path length formula, and a free boundary stem remains. ■

Working from the outside in, in this order, avoids creating situations where a boundary K_1 is changed to a K_2, when it should have been left available for an adjacent k-path. Figure 4.5 shows an example of a minimum dominating set for a tree with only k-paths of lengths 4t+1, 4t+2 and 4t+3.

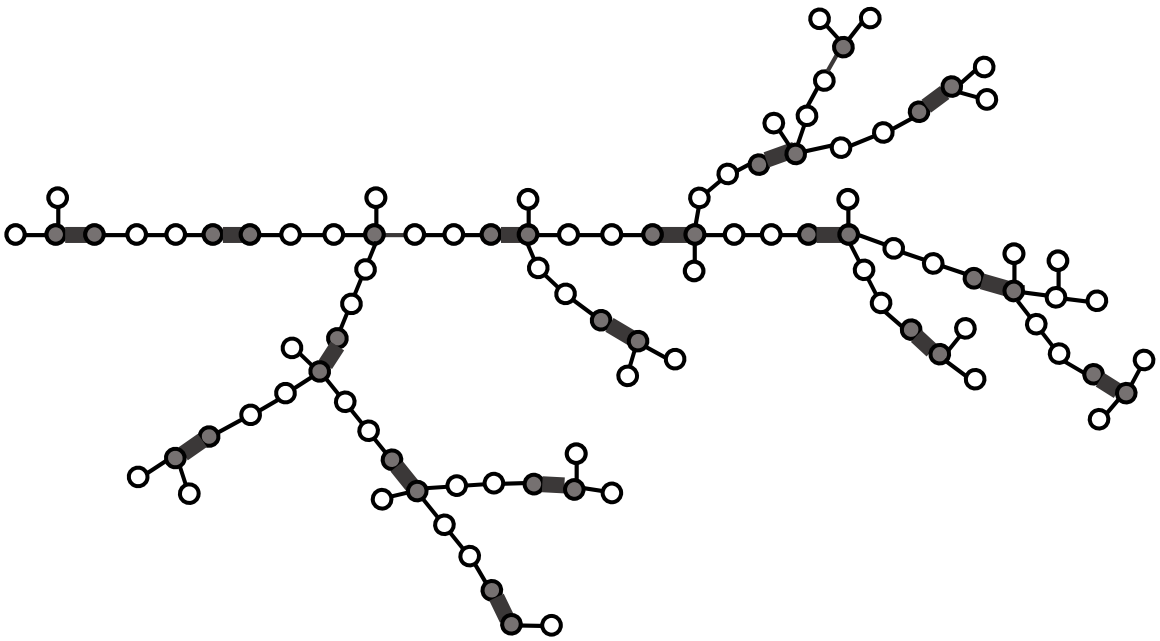

Figure 4.5: Example of a dominated tree of only k-paths of lengths 4t+1, 4t+2 and 4t+3.

Lemma 4.8 Let T be a tree where each high degree vertex is a stem, and such that whenever there are two k-paths of length 4t, there must be at least one k-path of length 4t+1 or 4t+2 between them. Then a minimum integrated dominating set can efficiently be found.

Proof: As shown earlier, the 4t k-paths need both boundary K_1's to be available to change to K_2's. If a tree T, has no 4t k-path, then Lemmas 4.6 and 4.7 apply. If not, there exists at least one 4t k-path in T.

Either the 4t k-path is at the outside end of a branch, in which case one of it's boundary K_1's is free, or it is adjacent to a string of 4t+1, 4t+2 or 4t+3 k-paths, which as shown earlier, can still allow for the joining K_1 to be free to change.

If two 4t k-paths are connected, with no 4t+1 or 4t+2 k-path between them, then an issue will arise as more K_1's will be needed to change to K_2's, than are available. However, if a 4t+1 or 4t+2 k-path is between them, it opens up one of the boundary K_1's for the 4t k-path, since they don't need their boundary K_1's changed.

For a minimum integrated dominating set for these trees, as with the trees in Lemma 4.7, K_1's are first placed on all stems, and since the 4t k-paths are top priority, all boundary K_1's of 4t k-paths are changed to K_2's appropriately. Next, starting at a path adjacent to a 4t k-path, the opposite K_1 is changed if necessary and guarding the k-paths follows a similar procedure as in Lemma 4.7 only here we are working backwards towards the outside of the tree. ■

Figure 4.6 shows two examples; one with no 4t+1 or 4t+2 k-path between pairs of 4t k-paths, and one with. It shows how without the 4t+1 or 4t+2 k-path, the second 4t k-path will not be able to use its boundary K_1 and so will not be able to be optimally guarded. In the second half of the figure, we can see how the 4t+1 or 4t+2 k-path helps with this situation.

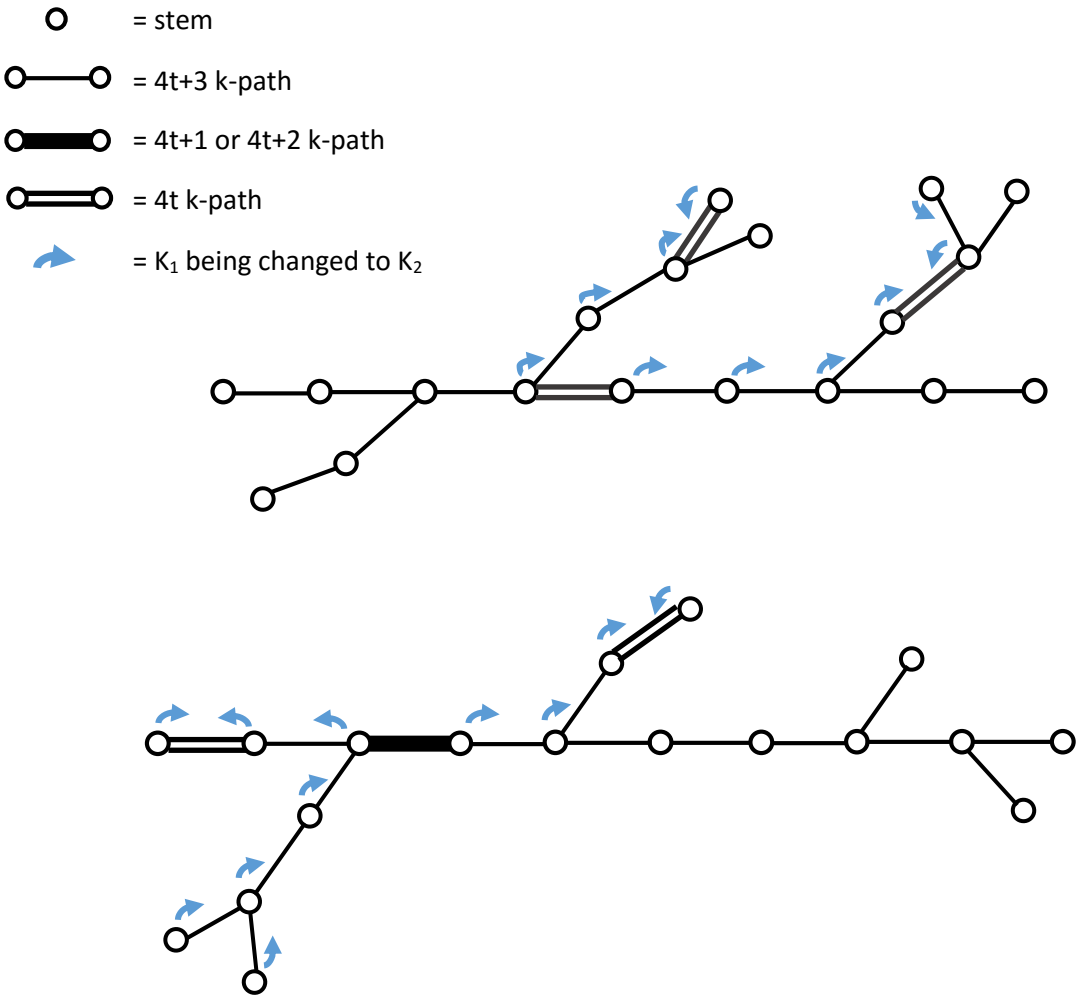

Figure 4.6: Examples of trees with and without a 4t+1 or 4t+2 k-path between pairs of 4t k-paths.

Lemma 4.9 Consider trees that can be constructed in the following manner:

First take a 4t-path and label the vertices in order from {1, …, n}. Attach 4t and 4t+1-arms to vertices in positions 0 and 1 mod 4, and 2 and 3 mod 4, respectively. Find a new 4t-path in the resulting tree and repeat the process by labelling the vertices in the new 4t path as

{1, …, m}, and attaching 4t and 4t+1-arms to vertices in positions 0 and 1 mod 4, and 2 and 3 mod 4, as before. This procedure can be continued until a desired tree is reached.

For trees satisfying this condition, a minimum integrated dominating set can efficiently be found.

Proof: Let P_a be a path of length 4t in T. Then P_a will be best dominated using t K_2's. Let $\{b_1, b_2, \ldots, b_m\}$ be the branches of length 4t+1 attached to P_a. As they are attached in positions 2 and 3 mod 4, this means that after K_2's are placed on P_a they will be attached to vertices that have guards on them. Thus, these guards will guard one vertex along each of the $\{b_1, b_2, \ldots, b_m\}$ branches, which will reduce their length to 4t, and thus they can also be guarded using t K_2's.

Let $\{c_1, c_2, \ldots, c_n\}$ be the 4t branches extending from P_a. As they are in positions 0 and 1 mod 4, after P_a is guarded with K_2's, they will be attached to vertices that are adjacent to guards. Thus, their unguarded length stays 4t, and they will use t K_2's.

Therefore, these trees are optimally guarded in this way, as the tree can be partitioned into the neighbourhoods of the K_2 guards. ■

Figure 4.7 illustrates an example of an integrated dominated tree with these conditions, showing two possibilities for the initial 4t-path.

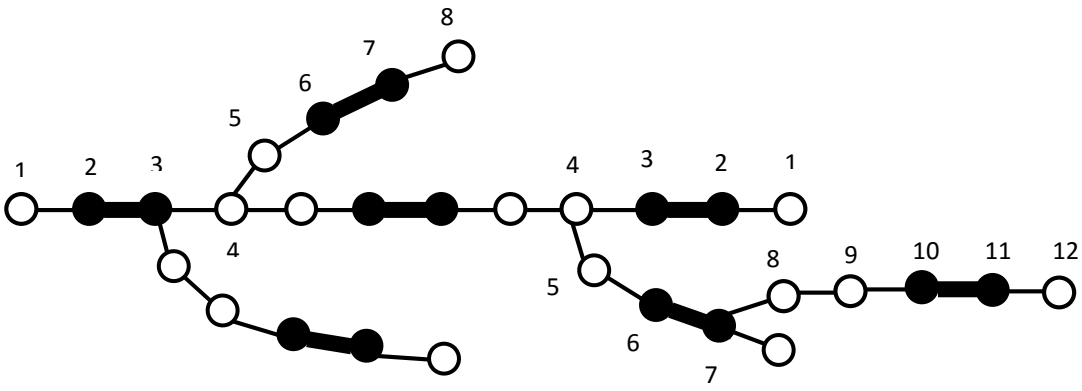

Figure 4.7: Example of a dominated tree with only 4t and 4t+1 branches.

Lemma 4.10 If a tree T consists of some high degree vertices such that each high degree vertex has a possibly empty set of 4t-arms and at least one 4t+3-arm attached, and each path between high degree vertices is of length 4t, then a minimum integrated dominating set can efficiently be found.

Proof: Let M = $\{m_1, m_2, \ldots, m_n\}$ represent the high degree vertices of T, satisfying the conditions given above.

Consider the subgraph consisting of $\{m_1\}$ and its k-arms only (paths leading to other sections of the graph are not considered). Looking at this subgraph by parts, it consists of 4t-paths, 4t+3 paths, plus the one vertex in the "middle", m_1.

Claim: For subgraphs like these, with at least one 4t+3 arm and all other arms of lengths 4t or 4t+3, a minimum dominating set involves placing a K_2 on a 4t+3-arm, adjacent to, but not on m_1.

Placing a K_2 on a vertex adjacent to, but not on m_1, on a selected 4t+3-arm guards m_1, and leaves the other arms of the subgraph with their original length unguarded. This allows the 4t-arms to be guarded optimally by K_2's and the selected 4t+3 arms to be guarded using t K_2's and one K_1 each. As for the selected arm, it is also maximally guarded as picking up m_1 creates an arm of length 4t+3 + 1, for which we would use t+1 K_2's.

If we were to place a K_2 adjacent to m_1 on a 4t-arm, the remaining 4t and 4t+3-arms follow the same result, but the selected 4t arm used, including m_1, would become of length 4t + 1 and would use t-2 K_2's and 3 K_1's, or t+1 guards in total. This would use one more guard used than when the K_2 was placed on the 4t+3-arm.

Note that placing a K_1 or K_2 directly on m_1 would increase the total number of guards used.

For 4t+3 lengths, increasing or decreasing the length of the arm by one does not change the number of guards used, it only changes the cost. This is because lengths 4t+3 and 4t+3 – 1 = 4t+2 use the same number of guards for the same value of t, specifically t+1 guards. Increasing 4t+3 by one gives us $4t^+$, and so would again use t+1 guards.

As an example, a length of 7 uses two guards. Decreasing it by one, we're left with 6 which also uses two guards, and increasing 7 by one gives us 8, which again uses two guards. Similarly, decreasing a 4t length by one changes the cost of the guards and not the quantity.

Therefore, having a K_1 on m_1, and so decreasing the lengths of all arms by one, doesn't affect the number of guards used for each arm. In fact, the subgraph in total,

guarded in this way, would use one more guard as all arms are using the same number of guards as before, but we're placing an extra guard on m_1.

In a similar way, having a K_2 on m_1 decreases one branch length by two, and all others by one. As we saw, decreasing these lengths by one doesn't affect the number of guards needed. The same is true for decreasing 4t and 4t+3 lengths by two vertices. Thus, they will still use the same number of guards, and so having an extra guard on m_1 increases the total number of guards used

Therefore, having a guard directly on m_1 is not the best option. In addition, having a K_2 adjacent to m_1 on a 4t branch is not optimal. Thus, having a K_2 adjacent to m_1 on a 4t+3-arm, and guarding the remaining arms based on the path length formulas from before gives an optimal solution.

At this point, we have dealt with the high degree vertices of our tree. Now, we see that between each of these high degree vertices is a path of length 4t that is not yet guarded. This can optimally be guarded using K_2's. ■

Figure 4.8 shows an example of the type of tree described here, minimally dominated using K_1's and K_2's.

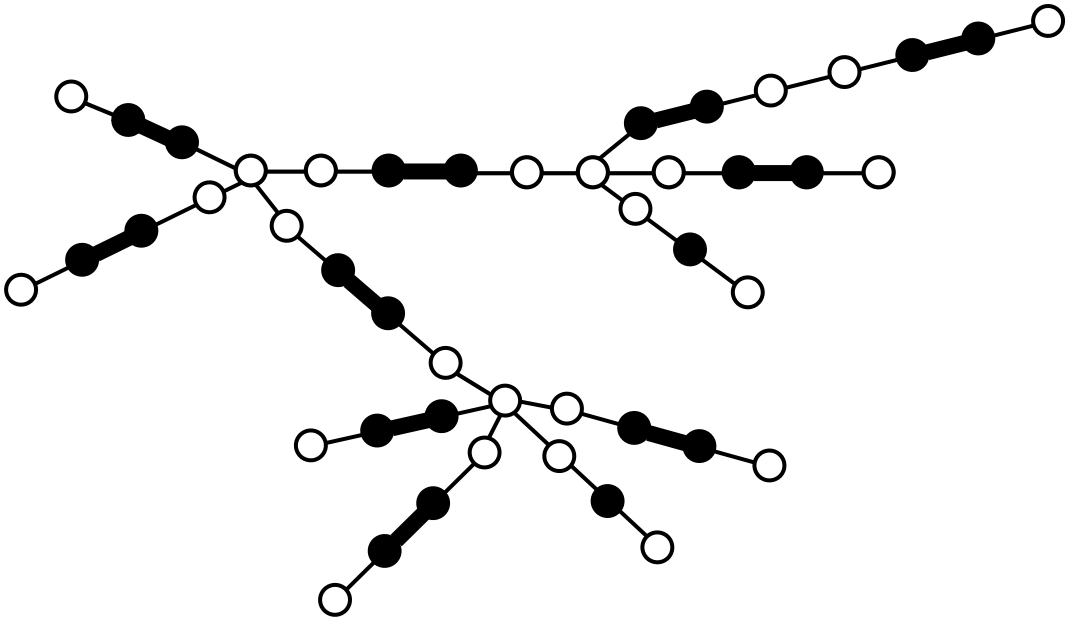

Figure 4.8: Example of an integrated dominated tree where each high degree vertex has 4t and 4t+3-arms, and each path between high degree vertices is of length 4t.

5 Conclusion

With this work, we hope to have given a good introduction to the notion of integrating different types of guards to minimally dominate a graph, focusing specifically on using K_1's and K_2's together.

Originally we had thought this problem, in regards to trees, might have been relatively easy since an algorithm for minimally dominating trees exists for both K_1 and K_2 guards, however it does not appear to be so simple.

Our first goal with this problem was to characterize subgraphs that we can show how to best dominate using K_1's and K_2's, such that given any graph, one could break it into sections and then guard each section separately, and in doing so dominate the entire graph. Section 3 discusses the subgraphs for which we were able to reduce and efficiently find a minimum integrated dominating set. However, there are still some cases that we have as yet been unable to solve. One such is similar to the graph in Lemma 4.11, but where the k-arms and paths between high degree vertices are of arbitrary lengths.

After attempting this, the next question that was asked was, "If we can't find a polynomial solution for an arbitrary graph, for which graphs can we find a solution?"

(relating back to Section 1.3: Some Techniques of Hard Problems). This question in turn led to a few results, with our characterized integrated dominated graphs found in Section 4. Although we have given a few types, it is anticipated more graphs can be dominated efficiently in this way. This is one aspect of the problem that is open for further research.

Another aspect of this problem that would be interesting to look at further, is if a solution to the problem could be performed in polynomial time. As dominating using K_1's and K_2's separately can be accomplished in this way, one would think that using both styles together could also be done in polynomial time. However, the answer to this might first require a solution to the problem, which of course would be ideal to find.

Another expansion to our problem would be to look at weighting the guards, in terms of cost, more efficiently. This stems back to the issue mentioned at the start, where although we want to find the solution with the fewest number of guards, choosing 99 K_2's over 100 K_1's does not seem to be so efficient. Finding a way to make the costs of the guards more practical and efficient would again be a useful extension to the problem.

One final idea, might be to look at which other forms of guards can be combined in dominating sets. From Section 1 we have seen that there are many different forms of domination, with many variations on the rules of the dominating set. Here we only use the two standard forms of guards, however many combinations could be possible, and might even lead to an efficient algorithm.

With this being said, there are many roads one could take when looking to expand on the problem. Here we give a few suggestions, however one could find new ways to develop or improve our results given here. We hope this introduction to integrated

domination using K_1's and K_2's has been an interesting read, and inspires future research into this topic.

6 References

[1] R. Book. Review: Michael R. Garey and David S. Johnson, Computers and Intractability: A guide to the theory of NP-completeness. Bull. Amer. Math. Soc. (N.S.), 3(2): 898-904, 1980.

[2] G. Chartrand and L. Lesniak. Graphs & Digraphs, Fourth Edition, Chapman & Hall/CRC, 2005.

[3] E. Cockayne, R. Dawes, and S. Hedetniemi. Total Domination in graphs. Networks, 10:211-219, 1980.

[4] E. Cockayne, B. Gamble, and B. Shepherd. An upper bound for the k-domination number of a graph. J. Graph Theory, 9:533-534, 1985.

[5] E. Cockayne, S. Goodman and S. Hedetniemi. A linear algorithm for the domination number of a tree, Inform.Process.Lett., 4:41-44, 1975.

[6] E. Cockayne and S. Hedetniemi. Towards a theory of domination, Networks 7, 247-61, 1977.

[7] C. Colbourn, P. Slater and L. Stewart. Locating-dominating sets in series-parallel networks. Congr. Numer., 56:135-162, 1987.

[8] M. Cozzens and L. Kelleher. Dominating cliques in graphs. Discrete Math., 86:145-164, 1990.

[9] C. De Jaenisch. Applications de l'analuse mathematique an Jen des Echecs. Petrograd, 1862.

[10] Jack Edmonds. Paths, Trees, and Flowers, Canad. J. Math. 17, 449-467, 1965.

[11] A. Finbow and B. Hartnell. A game related to covering by stars. Ars Combin., 16-A: 189-198, 1983.

[12] A. Finbow, B. Hartnell annd R. Nowakowski. A characterization of well-covered graphs of girth 5 or greater, Journal of Combinatorial Theory Series B (Impact Factor: 0.98), 57(1):44–68, 1993.

[13] J. Fink and M. Jacobson. n-domination in graphs. In Y. Alavi and A.J Schwenk, editors, Graph Theory with Applications to Algorithms and Computer Science, 283-300, (Kalamazoo, MI 1984), 1985. Wiley.

[14] S. Fitzpatrick. Aspects of domination and dynamic domination. Submitted in partial fulfillment of the requirements for the degree of doctor of philosophy at Dalhousie University, Halifax, Nova Scotia, August 1997.

[15] O. Goldreich. Computational Complexity: A Conceptual Perspective, Cambridge University Press, 2008.

[16] B. Hartnell and S. Fitzpatrick. Well paired-dominated graphs, J. Comb. Optim. 20, no. 2, 194-204, 2010.

[17] T. Haynes, S. Hedetniemi and P. Slater. Fundamentals of Domination in Graphs, Marcel Dekker, Inc., 1998.

[18] T. Haynes and P. Slater. Paired-domination in graphs. Networks, 32: 199-206, 1998.

[19] S. Hedetniemi, D. Jacobs, R. Laskar and D. Pillone. Open perfect neighborhood sets in graphs. Unpublished manuscript, 1997.

[20] S. Hedetniemi and R. Laskar. Bibliography on domination in graphs and some basic definitions of domination parameters. Departments of Computer Science and Mathematical Sciences, Clemson University, Clemson, SC 29631, USA, 1988. doi:10.1016/0012-365X(90)90365-O.

[21] M. Henning, O. Oellermann and H. Swart. Bounds on distance domination parameters. J. Combin. Inform. System Sci., 16:11-18, 1991.

[22] L. Kelleher and M. Cozzens. Dominating sets in social network graphs. Math. Social Sci., 16:267-279, 1988.

[23] D. König. Theorie der Endlichen and Unendlichen Graphen, Chelsea, New York, 1950.

[24] D. Kratsch. Finding dominating cliques efficiently, in strongly chordal graphs and undirected path graphs. Discrete Math., 86:225-238, 1990.

[25] V. R. Kulli. Theory of Domination in Graphs, VISHWA International Publications, 2010.

[26] R. Laskar, J. Pfaff, S. Hedetniemi and S. Hedetniemi. On the algorithmic complexity of total domination. SIAM J. Algebraic Discrete Methods, 5:420-425, 1984.

[27] M. Lewin. Matching-perfect and cover-perfect graphs. Israel J. Math. **18**, 345-347, 1974.

[28] O. Ore. Theory of Graphs. Amer. Match. Soc. Colloq. Publ., 38 (Amer. Math. Soc. Providence, RI) pp. 206-212, 1962.

[29] J. Pfaff, R. Laskar and S. Hedetniemi. Linear algorithms for independent domination and total domination in series-parallel graphs. Congr. Numer., 45:71-82, 1984.

[30] M. Plummer. Some covering concepts in graphs. J. Combin. Theory., 8:91-98, 1970.

[31] H. Qiao, L. Kang, M. Cardei and D. Du. Paired-domination of Trees, Kluwer Academic Publishers, 2003.

[32] P. Slater. R-domination in graphs. J. Assoc. Comput. Mach., 23:446-450, 1976.

[33] J. Staples. On some subclasses of well-covered graphs. J. Graph Theory **3**, 197-204, 1979.

[34] K. Sutner. Linear cellular automata and the Garden-of-Eden. Math. Intell., 12(2):49-53, 1989.

[35] A. Tucker. Applied Combinatorics, 5th Edition. John Wiley & Sons, Inc. N.Y., 2007.