Digraphs and $k$-Domination Models for Facility Location Problems in Road Networks: Greedy Heuristics

Lukas Dijkstra  
School of Mathematics, Cardiff University  
Cardiff, Wales, UK  
DijkstraL@cardiff.ac.uk

Andrei Gagarin  
School of Mathematics, Cardiff University  
Cardiff, Wales, UK  
GagarinA@cardiff.ac.uk

Padraig Corcoran  
School of Computer Science & Informatics, Cardiff University  
Cardiff, Wales, UK  
CorcoranP@cardiff.ac.uk  
Rhyd Lewis  
School of Mathematics, Cardiff University  
Cardiff, Wales, UK  
LewisR9@cardiff.ac.uk

ABSTRACT
We consider modelling the placement of refuelling facilities for alternative fuel vehicles in road networks by using directed graphs and $k$-dominating sets. The concept of a reachability digraph corresponding to a road network is introduced, and three greedy heuristics are proposed and experimentally tested to search for $k$-dominating sets in two types of digraphs, including the reachability digraphs of road networks. These simple and efficient heuristics show that refined greedy strategies usually provide better results for large as well as small digraphs, and their results are reasonably close to exact solutions for small digraphs. Combining the greedy strategies with some randomized heuristic ideas helps to improve the results even further in the case of digraphs associated with the road networks.

1 INTRODUCTION
1.1 Motivation
Dominating sets in simple graphs and networks have attracted a lot of attention from different perspectives, be they theoretical [1, 4, 12] or more applied [8, 19–21] in nature. However, directed graphs, or digraphs, which are more general abstract models in comparison to the simple graphs, are often overlooked. For example, the classic book on digraphs [2] does not pay much attention to dominating sets, and the classic book on dominating sets [12] does not pay much attention to digraphs. Digraphs offer advantages of more subtle modelling tools though, like representing one-way streets in road networks. Also, digraphs allow us to account for separate costs or differences in fuel consumption depending on which direction a road is travelled. These properties of digraphs are very useful when modelling road networks, a major area of application of graph theory [3, 5, 7, 8, 21]. On the other hand, in simple graphs it is not clear how to represent one-way streets or roads that are on an incline, causing fuel consumption to differ dramatically depending on whether a vehicle is going up- or downhill.

1.2 Basic Definitions, Notions, and Notation
A digraph $D$ is defined as $D = (V, A)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices and $A = \{e_1, e_2, \ldots, e_m\}$ is a set of ordered pairs of vertices called arcs. So, each arc $e \in A$ is of the form $e = (v_i, v_j)$ for some $i, j \in \{1, \ldots, n\}, i \neq j$. An arc $e = (v_i, v_j)$ as well as its inverse $e^{-1} = (v_j, v_i)$ may both be included in $A$ and, in this case, are treated as independent entities. Thus, $0 \leq m \leq n(n - 1)$. The out-neighbourhood of a vertex $v \in V$ is defined as the set $N^+(v) = \{u \in V \mid (v, u) \in A\}$, i.e. the set of vertices that are directly reachable from $v$ by traversing exactly one arc. On the other hand, the in-neighbourhood of $v$, $N^-(v) = \{u \in V \mid (u, v) \in A\}$, is the set of vertices that have an arc leaving them that leads to $v$. Additionally, the closed out-neighbourhood of $v$ is defined to be $N^*[v] = N^+(v) \cup \{v\}$. Similarly, the closed in-neighbourhood of $v$ is defined to be $N^*[v] = N^-(v) \cup \{v\}$. The out-degree of $v$ is $d^+(v) = |N^+(v)|$, and the in-degree of $v$ is $d^-(v) = |N^-(v)|$. These are the numbers of vertices directly reachable from $v$ and such that $v$ is directly reachable from them, respectively. The minimum out- and in-degrees of $D$ are denoted by $\delta^+ = \min\{d^+(v) \mid v \in V\}$ and $\delta^- = \min\{d^-(v) \mid v \in V\}$, respectively. These are the smallest degrees found across all vertices in the digraph.
Given a set of vertices $X \subseteq V$, a vertex $v$ is said to be covered by $X$ if $N^-[v] \cap X \neq \emptyset$, i.e. when $v$ is either in the set $X$ or can be directly reached via an arc from a vertex in $X$. The set of vertices covered by $X$ is denoted by $C(X) = \{v \in V \mid N^-[v] \cap X \neq \emptyset\}$. The set $X$ is called a dominating set of $D$ if $C(X) = V$, i.e. when every vertex of $D$ is either in $X$ or directly reachable from a vertex in $X$. More generally, for any integer $k \geq 1$, $v$ is said to be $k$-covered by $X$ if either $v \in X$ or $|N^-[v] \cap X| \geq k$. In other words, $v$ is either in $X$ or directly reachable by arcs from at least $k$ vertices in $X$. The set of vertices that are $k$-covered by $X$ in $D$ is denoted by $C_k(X)$, and $X$ is a $k$-dominating set of $D$ if $C_k(X) = V$. Note that, in these terms, covering by a set of vertices and a dominating set are simply the case of $k = 1$.

A $k$-dominating set $X$ of $D$ is minimal (by inclusion) if no vertex can be removed from $X$ without the resulting set losing the $k$-dominating set property, i.e. if we have $C(X \setminus \{v\}) \neq V$ for every $v \in X$. A $k$-dominating set $X$ of $D$ is a minimum $k$-dominating set if there does not exist a $k$-dominating set $Y$ of $D$ of a smaller size. The $k$-domination number of a digraph $D$ is the size of a minimum $k$-dominating set of $D$, which is denoted by $\gamma_k(D)$. Some basic theoretical results for the $k$-domination number of digraphs can be found in [16].

Thus, given a digraph $D$, we are interested in the problem of finding small-sized $k$-dominating sets in $D$, while using $\gamma_k(D)$ as a quality benchmark, whenever possible. Also, we want to find such sets of vertices in $D$ quickly.

2 HEURISTICS

Recent research focused on efficient bespoke heuristics to search for small $k$-dominating sets in simple graphs [5, 8]. Here we propose and consider three different greedy heuristic methods to tackle a similar problem in digraphs. These heuristics are called Basic Greedy (Algorithm 2), Deficiency Coverage Greedy (Algorithm 3), and Two-Criteria Greedy (Algorithm 4). A fourth heuristic method, relying on a combination of greedy and randomized ideas, is also proposed. Each of the heuristics starts by finding a $k$-dominating set of the digraph, which is usually not minimal (by inclusion). Therefore, at the end of the four main heuristics, an additional greedy heuristic is run to remove unnecessary vertices and to reduce the initially found set to a minimal $k$-dominating subset, or to check minimality of the initially found set. This Minimal $k$-Dominating Subset greedy heuristic is described by Algorithm 1.

2.1 Main Greedy Heuristics

An intuitive basic greedy strategy to find a $k$-dominating set in a simple graph is to start with an empty set $X$ and to add vertices into $X$, one at a time, by choosing iteratively a vertex with the most vertices in its closed neighbourhood that are not yet $k$-covered by $X$. This recursive procedure can be repeated until all vertices of the graph are $k$-covered by $X$, at which point $X$ is a $k$-dominating set. This strategy has been studied previously in the context of domination [1, 17] as well as $k$-domination [5, 8] in simple graphs. This basic greedy strategy generalizes to digraphs by checking specifically the closed out-neighbourhood of vertices at each step in iteration. This is described in the pseudocode of Algorithm 2.

### Algorithm 1: Minimal $k$-Dominating Subset

**Input:** A digraph $D = (V, A)$, an integer $k \geq 1$, a $k$-dominating set $X$ of $D$.

**Output:** A minimal $k$-dominating set $Y$ of $D$.

```
begin
  foreach $v \in X$ do
    Determine $x_v = |N^+(v) \setminus X|$
  end
  Initialize $Y = X$
  while $X \neq \emptyset$ do
    Find a vertex $v \in U = \arg\min_{u \in X} x_u$
    if $C_k(Y \setminus \{v\}) = V$ then
      Put $Y = Y \setminus \{v\}$
    end
    Put $X = X \setminus \{v\}$
  end
  return $Y$
end
```

### Algorithm 2: Basic Greedy

**Input:** A digraph $D = (V, A)$, an integer $k \geq 1$.

**Output:** A minimal $k$-dominating set $Y$ of $D$.

```
begin
  Initialize $X = \emptyset$
  while $C_k(X) \neq V$ do
    Find a vertex $v \in U = \arg\max_{u \in V \setminus X} |N^+[u] \setminus C_k(X)|$
    Put $X = X \cup \{v\}$
  end
  Find a minimal $k$-dominating set $Y \subseteq X$
  return $Y$
end
```

It is important to note that, in the case of $k > 1$, when searching for a $k$-dominating set of a (di)graph, vertices that are not yet $k$-covered by some vertex set $X$ can have different numbers of (in-)neighbours already in $X$. As a consequence, it can be more difficult to $k$-cover these vertices by their (in-)neighbours while expanding $X$ in the (di)graph. On the other hand, since including a vertex into $X$ results in $k$-covering this vertex regardless of how many (in-)neighbours the vertex has in $X$, it maybe more interesting to prioritize adding into $X$ the vertices that are not well $k$-covered yet to reduce the amount of vertices ((in-)neighbours) needed for their $k$-covering later.

Therefore, given a set of vertices $X \subseteq V$ of a digraph $D = (V, A)$ and an integer $k \geq 1$, the deficiency of a vertex $v \in V \setminus X$ is defined as $l_k(v, X) = \max\{0, k - |N^-[v] \cap X|\}$. This represents the amount of in-neighbours that are still needed to completely $k$-cover $v$ in the digraph. Algorithm 3, called Deficiency Coverage Greedy, is described below. It follows a modified greedy strategy of Basic Greedy of Algorithm 2. In contrast to Basic Greedy, Deficiency Coverage Greedy finds $k$-dominating sets by selecting vertices not only by their number of not $k$-covered out-neighbours, but also by the remaining deficiency of the vertex itself.
Another greedy strategy, introduced and computationally tested as a part of this research, can be considered as a refinement of Deficiency Coverage Greedy. In the Deficiency Coverage Greedy strategy, when several vertices can be used as the best candidates to be included into a set under construction, the algorithm chooses one of them uniformly at random. Instead, it is possible to make choice of the best candidate by considering the out-neighbours of each of these equally-ranked vertices.

Given a vertex \( v \in V \) of a digraph \( D = (V, A) \), we define the total out-neighbour in-degree of \( v \) to be \( f_D(v) = \sum_{u \in \text{Out}(v)} d^+(u) \). The additional greedy strategy uses the following heuristic assumption and observations. Since a vertex of low in-degree has fewer possible ways to be eventually \( k \)-covered in the digraph by its in-neighbours, if there is no efficient way to \( k \)-cover it, such a vertex is likely to be included in the \( k \)-dominating set in a later iteration, in particular, if its out-degree is much higher than its in-degree. Therefore, Algorithm 4, called Two-Criteria Greedy, prioritizes vertices \( v \) with a higher total out-neighbour in-degree \( f_D(v) \) in iteration. This is to discourage adding vertices with lower in-degree out-neighbours, because such out-neighbours are likely to be included themselves into the set under construction at a later point of time, which would reduce effectiveness of including the original vertex during the process. This Two-Criteria Greedy method is described in Algorithm 4.

Algorithm 3: Deficiency Coverage Greedy

**Input:** A digraph \( D = (V, A) \), an integer \( k \geq 1 \).

**Output:** A minimal \( k \)-dominating set \( Y \) of \( D \).

```
begin
    Initialize set \( X = \emptyset \)
    while \( C_k(X) \neq V \) do
        Find a set \( U = \arg \max_{u \in V \setminus X} |N^+(u) \setminus C_k(X)| + I_k(u, X) \)
        Select \( v \in U \) /* uniformly at random */
        Put \( X = X \cup \{v\} \)
    end
    Find a minimal \( k \)-dominating set \( Y \subseteq X \)
    return \( Y \)
end
```

The worst-case complexity analysis shows that all these greedy heuristics can be implemented to run in \( O(nm) \) time. This agrees with our implementation, for which the worst-case analysis provides a more detailed upper bound of \( O(n(n + m)) \).

2.2 Combining with a Randomized Heuristic

Although the algorithms above have some flexibility for the choice of a vertex at each iteration, they are very restrictive by their greedy selection nature. To fix this issue and to make them more flexible, one can try to use analytical tools and add more randomized components to the greedy strategies. In other words, we can combine the greedy strategies, for example, with a basic randomized technique.

A simple and efficient approach to make the greedy strategies above more flexible can consist in determining an initial random subset of vertices of a digraph for the greedy heuristics to start with (instead of an empty set). To do this in a more subtle and justified way, one can use a probabilistic method and corresponding analytical tools. Suppose we find an initial subset \( X \) of vertices for a \( k \)-dominating set by including (or not) each vertex of the digraph into \( X \) with some fixed probability \( p \) (respectively, \( 1 - p \)). One way to optimize this probability \( p \) is to use ideas from the probabilistic method.

The basic probabilistic method is a well-studied analytical tool [1, 9, 13], which can be used, for example, to find an upper bound for the domination number of a simple graph \( G = (V, E) \). It can be summarized as follows. Suppose we have some probability \( p \in [0, 1] \) to be specified or optimized later. First, find a random subset \( S \) of vertices of \( G \) by including each vertex of \( G \) into \( S \) independently with probability \( p \). Then, we have the subset \( R = V \setminus C(S) \) of vertices which are not covered by \( S \) in \( G \). Now, \( S \cup R \) is a dominating set of \( G \), as all the vertices not covered by \( S \) have simply been included into the set. The expected cardinality \( E(|S \cup R|) = E(|S|) + E(|R|) \) of this set can be computed explicitly in terms of \( p \) and is an upper bound for the domination number \( \gamma(G) \). The justification is straightforward: there must exist at least one dominating set obtained by using this method which has its cardinality at most the expected value. Since the expected cardinality can be considered as a function of \( p \), it can be optimized with respect to \( p \) to give the best possible upper bound for \( \gamma(G) \).

This approach has been generalized and applied to \( k \)-dominating sets in simple graphs. One of the best known results is as follows.

**Theorem 2.1** ([9]). Given a simple graph \( G = (V, E) \) with minimum vertex degree \( \delta \) and some integer \( k, 1 \leq k \leq \delta \),

\[
\gamma_k(G) \leq 1 - \left( \frac{\delta}{k-1} \right)^{\frac{1}{1 + \delta'}} n,
\]

where \( \delta' = \delta - k + 1 \).

The probability used to find this optimized upper bound in general simple graphs is \( p = 1 - \frac{1}{\sqrt[k-1]{\delta'}} \). However, as shown by the computational experiments in [8], this probability is too high for the reachability graphs of road networks. Therefore, instead of
the minimum vertex degree $\delta = \delta(G)$, alternative degree parameters of $G$, such as the mean and median vertex degrees, have been considered and used in the above algebraic expression.

Although the probabilistic method has been mainly used with simple graphs, it can be applied to digraphs as well. Lee [13] generalized the basic result of [1] to the domination number of digraphs, i.e. for the case of $k = 1$. We have obtained the corresponding result for the $k$-domination number of digraphs in general.

**Theorem 2.2.** Given a digraph $D = (V, A)$ with minimum in-degree $\delta^-$ and some integer $k$, $1 \leq k \leq \delta^-$,

$$\gamma_k(D) \leq \left(1 - \frac{\delta^-}{(k-1)^{\frac{1}{\delta^-}} \cdot (1 + \frac{1}{\delta^-})}\right)n,$$

where $\delta^- = \delta^- - k + 1$.

After optimization, the probability used to find this upper bound is $p = 1 - \frac{\delta^-}{(k-1)^{\frac{1}{\delta^-}} \cdot (1 + \frac{1}{\delta^-})}$. Therefore, we use this algebraic expression for probability $p$ to find an initial random subset of vertices in a digraph in an attempt to improve the results of greedy heuristics of Section 2.1. The complexity of finding an initial random subset of vertices in such a way is $O(n^2)$. It takes a linear time to decide with probability $p$ for each vertex $v_i$ whether to include or not $v_i$ in the subset, $i = 1, 2, \ldots, n$. However, computing $p$ involves computing the binomial coefficient, which can be done in $O(n^2)$ time in this case.

3 EXPERIMENTAL RESULTS

A number of computational experiments were run to test the heuristics from Section 2 and to compare and analyze the results. Two different types of digraphs were used in these experiments. Digraphs of the first type are randomly generated by using the so-called Erdős–Rényi (ER) random digraph model. This consists in taking a set of vertices, and, by using some fixed probability $p$, independently for each ordered pair of vertices, we decide whether the corresponding arc is in the digraph or not. Note that the two possible arcs between a pair of vertices are considered separately and therefore included or not into the digraph independently from each other.

The second type of digraphs are so-called reachability digraphs derived from actual road networks. A similar concept of a reachability graph is defined in [8] for simple graphs. In the reachability digraph model, vertices represent some locations in the road network. An arc from one vertex to another is included into the reachability digraph if it is possible to travel from the location corresponding to the first vertex to the location of the other vertex within a certain predefined road distance. The maximum travelling distance to have an arc is called the reachability radius ($r$) in the road network. In comparison to the ER random digraphs, the reachability digraphs are more similar to simple graphs, because many streets support two-way traffic. However, although most connections in a reachability digraph are two-way, there is still a non-negligible number of one-way connections that would be ignored in a simple graph model. This is illustrated in Figure 1, where the arcs originating from the blue vertex are leading only to the red vertices.

![Figure 1: A vertex (blue) and its out-neighbours (red) in a reachability digraph.](image)

Before running computational experiments on large digraph instances, we considered small size digraphs to be able to obtain some exact solutions. This allows us to compute $\gamma_k(D)$ to compare the greedy heuristics results. The exact deterministic solutions were obtained by solving an integer-linear programming (ILP) formulation of the problem by using Gurobi 10.0.1 [11]. We used the following ILP formulation:

$$\text{minimize } z(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n} x_i$$

subject to:

$$kx_i + \sum_{v_j \in N^+(v_i)} x_j \geq k, \quad i = 1, \ldots, n$$

$$x_i \in \{0, 1\}, \quad i = 1, \ldots, n,$$

where $x_i$ is a binary variable indicating whether vertex $v_i$ of the digraph is included in the $k$-dominating set or not, $i = 1, \ldots, n$. When it was not possible to solve the problem of computing $\gamma_k(D)$ in a reasonable amount of time, the generic ILP solver was run as an alternative heuristic solver using a substantial amount of CPU time resources.

We have considered $k$-domination for $k = 1, 2, 4, 8$. For some small-size digraphs, the deterministic method (ILP) was able to return an optimal solution within a reasonable timeframe for lower values of $k$, but started experiencing infeasibly large runtimes for higher values of $k$. Therefore, a time limit of 24 hours was imposed and, if it was reached, the best solution found so far (i.e. heuristic) would be recorded instead of the exact solution. The experiments were conducted by using C++ on a PC with a 3.00 GHz Intel Core i5.
Digraphs and \(k\)-Domination Models for Facility Location Problems in Road Networks: Greedy Heuristics

INOC 2024, March 11 - 13, 2024, Dublin, Ireland

We constructed digraphs corresponding to road networks by using OpenStreetMap (OSM) geographic information system data [15]. The corresponding road networks are comprised of all roads contained within a square box, the center of which is the Birmingham New Street train station in the United Kingdom (exact coordinates: 52.478691, -1.89984). The digraphs for the small-scale experiments are given by the box side-lengths of 1, 1.25, 1.5, 1.75, and 2 kilometers, with reachability radii of \(r = 300, 325, 350, 375\), and 400 meters, respectively. The large size digraphs are given by the box side-lengths of 10, 20, 30, 40, and 50 kilometers, with the reachability radii of 3, 4, 5, 6, and 7 kilometers, respectively. An experiment on a digraph corresponding to a road network of the box side-length of 60 kilometers and with the reachability radius of 8 kilometers was attempted, but the computer ran out of memory. Some of the results of these computational experiments are presented in Tables 3 and 4 below.

Similarly to the small Erdős–Rényi digraphs, Table 3 shows that the solution quality of the proposed greedy heuristics is comparable to the exact or heuristic ILP solutions after running the generic ILP solver on small reachability digraphs for a much longer time (one to three orders of magnitude more time to obtain exact ILP solutions, and five to six orders of magnitude more time to obtain alternative heuristic solutions). Also, Table 3 shows that, for the small reachability digraphs, when \(k > 1\), TCG usually provides better results than the other two greedy heuristics, and the advantages of DCG and TCG over BG become more visible for the larger values of \(k\). For the large reachability digraphs, Table 4 shows that TCG provides the best results for all but two problem instances (out of twelve), which are better solved by DCG. BG is still competitive for \(k = 1\), but for larger values of \(k > 1\), the advantages of DCG and TCG are more visible again. Notice that, for \(k = 1\), DCG would normally produce the same results as BG (TCG has the secondary selection criterion, which comes into play even when \(k = 1\)). However, the random choice of a vertex among the equally most suitable candidates in iteration of DCG produces slightly different from BG results and introduces the option of running the algorithm several times to potentially obtain better results. The runtimes of greedy heuristics on the same digraph instance are always comparable, and within a reasonable time limit (less than 30 min for the reachability digraph on 225289 vertices).

**3.1 Erdős–Rényi digraphs**

All of the Erdős–Rényi digraphs used in these experiments were generated using arc inclusion probability of \(p = 0.1\). The digraphs for the small-scale experiments had \(n = 100, 150,\) and 200 vertices, whilst the large-size digraphs contained 25, 50, 75, and 100 thousand vertices. An experiment on a digraph with 125 thousand vertices was also attempted, but the computer ran out of memory. Some of the results of these experiments are presented in Tables 1 and 2.

Table 1: \(k\)-Dominating sets in small Erdős–Rényi digraphs.

<table>
<thead>
<tr>
<th>(n)</th>
<th>(k)</th>
<th>(ILP)</th>
<th>(BG)</th>
<th>DCG</th>
<th>TCG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Size</td>
<td>Time (s)</td>
<td>Size</td>
<td>Time (s)</td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td>12</td>
<td>0.00057</td>
<td>12</td>
<td>0.00057</td>
</tr>
<tr>
<td>2</td>
<td>20</td>
<td>1.98</td>
<td>0.00067</td>
<td>24</td>
<td>0.00073</td>
</tr>
<tr>
<td>36</td>
<td>4.52</td>
<td>0.0009</td>
<td>35</td>
<td>0.00091</td>
<td>40</td>
</tr>
<tr>
<td>64</td>
<td>3.46</td>
<td>0.0011</td>
<td>66</td>
<td>0.0012</td>
<td>66</td>
</tr>
</tbody>
</table>

Table 2: \(k\)-Dominating sets in large Erdős–Rényi digraphs.

<table>
<thead>
<tr>
<th>(n)</th>
<th>(k)</th>
<th>(BG)</th>
<th>DCG</th>
<th>TCG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Size</td>
<td>Time (s)</td>
<td>Size</td>
</tr>
<tr>
<td>50,000</td>
<td>1</td>
<td>52</td>
<td>21.95</td>
<td>51</td>
</tr>
<tr>
<td>2</td>
<td>73</td>
<td>23.27</td>
<td>72</td>
<td>21.68</td>
</tr>
<tr>
<td>4</td>
<td>105</td>
<td>23.21</td>
<td>104</td>
<td>21.65</td>
</tr>
<tr>
<td>8</td>
<td>165</td>
<td>23.72</td>
<td>164</td>
<td>23.16</td>
</tr>
<tr>
<td>100,000</td>
<td>1</td>
<td>58</td>
<td>1654</td>
<td>58</td>
</tr>
<tr>
<td>2</td>
<td>78</td>
<td>1470</td>
<td>78</td>
<td>1422</td>
</tr>
<tr>
<td>4</td>
<td>113</td>
<td>1321</td>
<td>113</td>
<td>1576</td>
</tr>
<tr>
<td>8</td>
<td>174</td>
<td>1574</td>
<td>173</td>
<td>1646</td>
</tr>
</tbody>
</table>

4 CONCLUDING REMARKS

In this research, we have considered and accentuated using digraphs for modelling problems in road networks, introduced the concept of a reachability digraph corresponding to a road network, proposed modelling and optimization of facility locations in road networks by considering \(k\)-dominating sets in digraphs. By refining some greedy criteria, we have devised and computationally tested three different greedy heuristics, shown and discussed their performance with respect to some exact (or near-exact) solutions and each other by using two types of digraphs. To make the greedy heuristics more flexible and to improve their performance further, some randomization ideas are proposed as well.

Current and future research will focus on the randomization techniques to make these greedy heuristics more flexible and effective, and to be able to improve the obtained results for large-scale digraphs efficiently. We also plan to consider more subtle domination models in digraphs and more involved heuristic solution...
strategies, for example, applications and modifications of the local search. To help with exact solutions for small size problem instances in digraphs, we plan to consider devising customized deterministic algorithms. Notice that some recent research (see [14]) focused on greedy heuristics to search for small weight dominating sets in vertex-weighted digraphs.

ACKNOWLEDGMENTS
Lukas Dijkstra acknowledges funding from the Maths DTP 2020, EPSRC grant EP/V520159/1.

REFERENCES