Anomalous Behaviour of Dual-Based Heuristics

Trivikram Dokka* Adam N. Letchford* M. Hasan Mansoor*

January 2018, Revised June 2019

Abstract

Some popular heuristics for combinatorial optimisation start by constructing a feasible solution to a *dual* of the problem. We show that such *dual-based* heuristics can exhibit highly counter-intuitive behaviour. In particular, for some problem classes, solving the dual exactly invariably leads to much worse primal solutions than solving the dual with a simple greedy heuristic. We provide a tentative explanation for this phenomenon, based on the concept of primal degeneracy. We use the *simple plant location* and *set covering* problems as examples

Keywords: Heuristics, dual ascent, facility location, set covering.

1 Introduction

Many important combinatorial optimisation problems arising in practice are $\mathcal{NP}$-hard. Although highly effective exact solution methods have been developed for $\mathcal{NP}$-hard problems (see, e.g., [18, 34]), large-scale instances can still pose a challenge for exact methods. In such cases, one must resort to *heuristics* (see, e.g., [14]).

This paper is concerned with what we call *dual-based* heuristics. These heuristics start by constructing a feasible solution to some kind of dual problem (such as a linear programming dual or a Lagrangian dual). Once a dual solution is obtained, they then attempt to use information from the dual solution (such as reduced costs) to guide a primal heuristic. In fact, they usually generate a *sequence* of dual solutions, which then yields a sequence of primal solutions. At the end, one can just pick the best primal solution found during the course of the procedure.

Dual-based heuristics can be regarded as primitive examples of what are now called *matheuristics*, by which is meant heuristics that draw on concepts from the traditional mathematical programming literature (see,

*Department of Management Science, Lancaster University, Lancaster LA1 4YX, UK. E-mail: {T.Dokka,A.N.Letchford,H.Mansoor}@lancaster.ac.uk

e.g., [39]). In comparison with other matheuristics, however, dual-based heuristics have two very nice features. The first is that their running times and memory requirements tend to be bounded by a low-order polynomial in the instance size. As a result, they can easily be applied to large-scale instances. The second is that they yield both lower and upper bounds on the optimal objective value. If these bounds are close, one has the reassurance that the primal solution is of high quality.

Dual-based heuristics have proven to be highly successful on many well-known problems, including, e.g., the *simple plant location* [9, 12, 21], *set covering* [3, 7, 15], *Steiner tree* [41, 46], *uncapacitated network design* [2], *generalised assignment* [2] and *set partitioning* problems [13, 25]. More recently, researchers have begun looking into the possibility of speeding up dual-based heuristics using parallel processors (e.g., [19]).

A natural question, however, is whether there really is any useful information to be extracted from "good" dual solutions, or whether a random dual solution would do just as well. A partial answer to this question is that some dual-based heuristics have been shown to be *approximation algorithms* (see, e.g., [45]). Roughly speaking, this means that there is a bound on their worst-case error. Another partial answer is given in [47], in the context of the set covering problem (SCP). They show empirically that, if a dual solution is optimal, then the variables with zero reduced cost have a high probability of belonging to an optimal solution.

In this paper, we continue to study this question from an empirical (and statistical) viewpoint. We use the SCP and the simple plant location problem (SPLP) as test cases. For each problem, we consider several ways to construct dual solutions, and conduct extensive computational experiments. It turns out that dual-based heuristics can exhibit highly counter-intuitive behaviour. In particular, in the case of the SPLP, solving the dual exactly invariably leads to much worse primal solutions than solving the dual with a simple greedy heuristic. We provide a tentative explanation of this phenomenon, based on the presence or absence of *primal degeneracy.*

The paper has a very simple structure. In Section 2, we review the relevant literature. Sections 3 and 4 are devoted to the SPLP and SCP, respectively. Concluding remarks are made in Section 5. Throughout, we assume that the reader is familiar with basic concepts of linear and integer programming (see, e.g., [18] for a fine treatment).

2 Literature Review

We now briefly review the relevant literature. We recall the basics of dual-based heuristics in Subsection 2.1. In Subsections 2.2 and 2.3, we cover applications of the heuristics to the SPLP and SCP, respectively.

2.1 Dual-based heuristics

Suppose we have formulated a combinatorial optimisation problem as an *integer linear program* (ILP) of the form

$$\min\left\{c^T x :\ Ax \geq b,\ x \in \mathbb{Z}_+^n\right\},$$

where $A \in \mathbb{Q}^{m\times n}$ and $b \in \mathbb{Q}^m$. The dual of the LP relaxation of this ILP is

$$\max\left\{b^T \pi :\ A^T\pi \leq c,\ \pi \in \mathbb{R}_+^m\right\}.$$

Any feasible solution π to this dual provides a lower bound for the original problem. Thus, if we wish to obtain a lower bound quickly, we can solve the dual approximately using some kind of heuristic.

In *dual ascent*, one starts with all π variables set to small values, and then iteratively increases the value of a π variable, until no more can be increased without violating dual feasibility [12, 21]. If desired, one can attempt to improve the dual solution further by applying local search. This is called *dual adjustment* in [21].

Now, for a given primal variable x_j and a given dual solution π, consider the following quantity:

$$\bar{c}_j(\pi) \,=\, c_j - \sum_{i=1}^{m} \pi_j A_{ij}.$$

We call this the *approximate reduced cost.* Intuitively speaking, if π is a near-optimal dual solution, one might expect variables with small approximate reduced cost to have a high probability of belonging to an optimal solution of the original ILP. This leads to the idea of an integrated scheme in which the approximate reduced costs are used to guide a primal heuristic [12, 21]. This is what we mean by a *dual-based heuristic.*

For some problems, *Lagrangian relaxation* (LR) can provide an attractive alternative to dual ascent. Consider an ILP of the form:

$$\min\left\{c^T x :\ Ax \geq b,\ Cx \geq d,\ x \in \mathbb{Z}_+^n\right\},$$

where $A \in \mathbb{Q}^{m\times n}$ and $C \in \mathbb{Q}^{q\times n}$. Relaxing the constraints $Cx \geq d$, with a vector $\lambda \in \mathbb{R}^q$ of *Lagrangian multipliers*, we obtain

$$\min\left\{c^T x + \lambda^T(d - Cx) :\ Ax \geq b,\ x \in \mathbb{Z}_+^n\right\},$$

The solution of this relaxed problem yields a lower bound (see, e.g., [24, 27, 30]). Moreover, for a given primal variable x_j and a given λ, the quantity

$$\bar{c}_j(\lambda) \,=\, c_j - \sum_{i=1}^{q} \lambda_j C_{ij}$$

can also be viewed as an approximate reduced cost (see, e.g., [15]).

2.2 Application to the SPLP

In the SPLP (a.k.a. the *Uncapacitated Facility Location Problem*), there is a set I of facilities and a set J of clients. For any $i \in I$, it costs f_i to open facility i. For any $i \in I$ and $j \in J$, it costs c_{ij} to serve client j from facility i. One must decide which facilities to open, and assign each client to an open facility, at minimum cost. Balinski [5] formulated the SPLP as the following 0-1 LP:

$$\begin{aligned}
\min \quad & \textstyle\sum_{i \in I} f_i y_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} && && (1)\\
\text{s.t.} \quad & \textstyle\sum_{i \in I} x_{ij} = 1 && (\forall j \in J) && (2)\\
& y_i - x_{ij} \geq 0 && (\forall i \in I, j \in J) && (3)\\
& x_{ij} \in \{0,1\} && (\forall i \in I, j \in J) && (4)\\
& y_i \in \{0,1\} && (\forall i \in I). && (5)
\end{aligned}$$

Here, x_{ij} indicates whether client j is assigned to facility i, and y_i indicates whether facility i is opened.

The LP relaxation of this formulation often gives very tight lower bounds [1, 40]. On the other hand, the constraints (3) cause massive primal degeneracy, which makes the LP relaxation hard to solve (see, e.g., [44]).

A dual ascent approach was proposed in [12, 21]. They start by showing that the dual of the LP relaxation can be written as:

$$\begin{aligned}
\max \quad & \textstyle\sum_{j \in J} v_j && &&\\
\text{s.t.} \quad & \textstyle\sum_{j \in J} \max\{0, v_j - c_{ij}\} \leq f_i && (\forall i \in I) && (6)\\
& v_j \geq 0 && (\forall j \in J), &&
\end{aligned}$$

where $v \in \mathbb{R}^{|J|}$ is the vector of dual variables for constraints (2). They then proceed as follows. First, set each v_j to the cost of assigning client j to its closest facility (i.e., the smallest c_{ij} value over all $i \in I$). Then, examine each client $j \in J$ in turn, and increase v_j to the next smallest c_{ij} value, or, if this renders the dual solution infeasible, by as much as possible while maintaining feasibility. Repeat until no more increases are possible.

One can check that, for a given v, the approximate reduced costs of the facilities are just the slacks of the constraints (6). Moreover, for any pair i, j, the quantity $\max\{0, v_j - c_{ij}\}$ can be viewed as an estimate of the dual price of the constraint (3). This led Erlenkotter to propose a primal heuristic, in which one iteratively opens facilities whose approximate reduced cost is zero until, for each client $j \in J$, there is an open facility such that $v_j - c_{ij} = 0$.

One can obtain improved dual solutions by applying local search (see, e.g., [21, 32, 33, 36, 42]). This typically yields improved primal solutions too. An analogous Lagrangian approach, in which constraints (2) are relaxed, was studied in [9, 26]. See also [11, 29, 38] for related approaches.

2.3 Application to the SCP

In the SCP, we are given positive integers m and n, a family of sets $S_1, \ldots, S_n \subset \{1, \ldots, m\}$, and a cost c_j for $j = 1, \ldots, n$. The task is to find a minimum-cost collection of sets such that each member of $\{1, \ldots, m\}$ is contained in at least one set in the collection. The SCP can be formulated as a 0-1 LP of the form:

$$\min \left\{ c^T x : \ Ax \geq e_m, \ x \in \{0,1\}^n \right\}, \tag{7}$$

where x_j indicates whether the jth set has been selected, the columns of $A \in \{0,1\}^{mn}$ encode the sets, and e_m is the all-ones vector of order m.

Since the upper bounds of one on the x variables are redundant, the dual of the LP relaxation can be written in the simple form:

$$\max \left\{ e_m^T \pi : \ A^T \pi \leq c, \ \pi \in \mathbb{R}_+^m \right\}.$$

Balas and Ho [4] were the first to apply dual ascent to the SCP. They begin with π set to the all-zeroes vector. They then sort the rows of A in non-decreasing order of density (i.e., number of '1's), and then go through the sorted list, pushing each π variable up to its maximum possible value. To construct primal solutions, they set to one all x variables for which $\bar{c}_j(\pi) = 0$, and then drop variables, if necessary, to make the cover minimal. Beasley [6] followed a similar scheme.

Fisher & Kedia [25] improved the scheme in two ways. First, they added a local search phase to improve the dual solution. Second, they used an improved primal heuristic, which proceeds as follows. Start with an empty cover. For $j = 1, \ldots, n$, let $\kappa(j) \subset \{1, \ldots, m\}$ be the set of *currently uncovered* rows that have a 1 in column j, and compute $\tilde{c}_j = c_j - \sum_{i \in \kappa(j)} \pi_i$. (Note that $\tilde{c}_j$ lies between the original cost c_j and the approximate reduced cost $\bar{c}_j(\pi)$.) Set to one the x variable that minimises $\tilde{c}_j / |\kappa(j)|$. Recompute the $\tilde{c}_j$, and repeat until a cover is obtained. At the end, the cover is again made minimal if necessary.

Lagrangian relaxation has been applied to the SCP by many authors (e.g., [3, 4, 6, 7, 10, 15, 17, 31]). In all of those papers, all of the linear constraints are relaxed, and the subgradient method is used to get good multipliers. To construct primal solutions, most of those authors simply set x variables to one in non-decreasing order of $\bar{c}_j(\lambda)$ until a cover is obtained, and then make the cover minimal, if necessary, by setting a few variables back to zero. To date, the best-performing Lagrangian scheme is that of Caprara *et al.* [15], who used a clever technique to improve convergence in the subgradient method, sophisticated rules for fixing primal variables, and a primal heuristic analogous to that of [25] (mentioned above).

An initial study of the value of dual information for the SCP has been made in [47]. They give evidence that, when the dual solution is *optimal*, variables with zero reduced cost have a high probability of belonging to an optimal solution.

3 The Simple Plant Location Problem

This section is concerned with the SPLP. In Subsections 3.1 and 3.2, we present deterministic and randomised procedures, respectively, for generating dual and primal solutions. In Subsections 3.3 and 3.4, we present and analyse computational results obtained with the deterministic and randomised procedures, respectively.

3.1 Deterministic procedures

Dual ascent is fast and simple, but the dual solution obtained is typically sub-optimal. We found it useful to compute an optimal dual solution as well. To do this, we simply solve the LP relaxation of (1)–(5). (This is easy using modern LP solvers, even for quite large instances.)

We also experimented with using Lagrangian relaxation instead of dual ascent, as in [9]. The idea is to relax the constraints (2), and then use the subgradient method to generate a near-optimal vector of Lagrangian multipliers, from which approximate reduced costs can be computed. Unfortunately, we did not find the results illuminating, because the multipliers and primal solutions encountered depended heavily on the starting point, the step size, the stopping rule, and so on.

To generate a primal solution from a given dual solution, we use the *dual-based drop* heuristic that was presented in [38]. This heuristic, which is a modified version of the classical *drop* heuristic of Feldman *et al.* [22], is described in Algorithm 1. In our experience, the heuristic tends to perform slightly better than Erlenkotter's primal heuristic. Moreover, it is fast. (Specifically, it can be implemented to run in $O(mn \log m)$ time, where m is the number of facilities and n the number of clients.)

In order to provide a benchmark, we also consider a primal heuristic that does *not* exploit dual information. This heuristic is identical to Algorithm 1, except that we sort the facilities in non-increasing order of cost f_i rather than $\bar{f}_i$. We call this latter heuristic the *cost-based* drop heuristic.

3.2 Randomised procedures

The algorithms presented in the previous subsection yield only a small number of dual and primal solutions. Since we are interested in exploring potential correlations between the quality of dual and primal solutions, we really want many solutions of each type. This led us to devise randomised versions of the algorithms, which produce many solutions, rather than just one. After trying several alternative ways to generate dual solutions, we settled on the following two methods.

1. *Randomized Dual Ascent.* As already noted by Erlenkotter [21], the precise dual vector obtained via dual ascent depends on the order in

Algorithm 1: Dual-Based Drop Heuristic for SPLP

```
input  : positive integers $m$, $n$, cost vectors $f \in \mathbb{Z}_+^m$, $c \in \mathbb{Z}_+^{mn}$,
         dual solution $v \in \mathbb{Z}_+^n$.
Temporarily open all facilities;
Temporarily assign each client to the nearest facility;
Let $U$ be the cost of the initial solution;
for all $i \in I$ do
    Let $\bar{f}_i = f_i - \sum_{j \in J} \max\{0, v_j - c_{ij}\}$;
end
Sort the facilities in non-increasing order of $\bar{f}_i$ (breaking ties at
  random);
for $i = 1$ to $m$ do
    Let $k$ be the $i$th facility in the sorted list;
    Let $\Delta$ be the increase in total cost that would be incurred by
      closing facility $k$ and re-assigning each of its clients to the next
      nearest open facility;
    if $\Delta < 0$ then
        Close facility $k$ and re-assign its clients;
        Set $U := U + \Delta$;
    end
end
output: SPLP solution and associated upper bound $U$.
```

which the clients are considered. So we simply run dual ascent a large number of times, with the clients sorted in a random order in each major iteration.

2. *Perturbed LP*. We solve the LP relaxation of (1)–(5), but add a small random number, uniformly distributed in $[-0.25, 0.25]$, to the right-hand side of each constraint (2). The effect of the perturbation is to make some of the v_j more "attractive" than others in the dual. For the instances that we tested, 0.25 was the smallest value that enabled us to generate a reasonable number of distinct dual solutions. (We will see in Subsection 3.4 that those dual solutions were all optimal or near-optimal.)

In order to provide a benchmark, we also designed a randomised version of the cost-based drop heuristic, that we described in the previous subsection. Instead of selecting the next facility in the sorted list as the next candidate to be dropped, we select a facility at random from the next five facilities in the list. (Five was the smallest value that led to a diversity of primal solutions.)

3.3 Results with deterministic procedures

An extensive collection of benchmark SPLP instances is available here:

`http://www.mpi-inf.mpg.de/departments/d1/projects/benchmarks/UflLib/`

After some experimentation, we selected four families of instances:

- `K-median` instances, constructed as in Anh *et al.* [1]. In these instances, the facility and client locations are points in the unit square. For $m = n = 250$, we created five instances with large facility costs (`Kmed-L`), five with medium facility costs (`Kmed-M`) and five with small facility costs (`Kmed-S`). These instances have small integrality gaps, and we were able to find the optimal solutions easily using `CPLEX`.

- The `M*` instances of Kratica *et al.* [37]. In these instances, there is a negative correlation between the facility costs and assignment costs. They are hard to solve and have fairly large gaps. We consider only the 15 smallest ones, for which optimal solutions are known (see [42]). They have $m = n \in \{100, 200, 300\}$.

- The `KG` instances. These were created by Ghosh [28] using a similar scheme to that of Koerkel [36]. They too are hard to solve and have fairly large gaps. They come in two types, symmetric and asymmetric. We denote these by `KG-S` and `KG-A`, respectively. We selected the 30 smallest ones, for which optimal solutions are known (see, e.g., [23, 42]). They have $m = n = 250$.

Table 1: SPLP: Average percentage gaps with deterministic procedures.

		lower bounds		upper bounds			
Set	$m = n$	ascent	LP	ascent	simplex	barrier	cost-based
`Kmed-L`		0.54	0.00	2.09	1.97	0.36	11.8
`Kmed-M`	250	0.30	0.00	0.72	0.82	0.00	7.17
`Kmed-S`		0.00	0.00	0.02	0.04	0.00	0.03
`MO`	100	6.06	2.85	2.07	21.4	0.86	0.83
`MP`	200	7.27	4.20	2.64	20.7	0.21	1.31
`MQ`	300	7.21	4.04	1.47	23.7	0.28	1.07
`KG-S-A`		0.31	0.13	0.14	0.94	0.11	0.47
`KG-S-B`	250	1.48	0.96	0.64	3.42	0.37	1.35
`KG-S-C`		4.31	3.29	0.78	7.34	0.49	2.27
`KG-A-A`		0.30	0.14	0.16	0.86	0.11	0.48
`KG-A-B`	250	1.50	0.98	0.56	3.34	0.31	1.33
`KG-A-C`		4.11	3.13	0.79	8.08	0.67	2.32
`Gap-C`	100	68.8	28.2	43.1	41.0	32.0	40.9

- The instances from Kochetov & Ivanenko [35]. They only have $m = n = 100$, but they are designed to have very large integrality gaps (over 25% in all cases). There are 90 instances in total, and optimal solutions are now known for all of them. We selected the 30 hardest ones, called `Gap-C` instances.

We implemented the lower- and upper-bounding procedures in C, and we used the `CPLEX` callable library (v. 12) to solve all LP relaxations. Out of interest, we computed dual solutions using both simplex and barrier methods. (In the case of the barrier method, we switched off "crossover", to ensure that we found a dual solution that lies near the centre of the optimal face in the dual.)

For each instance and each procedure, we computed the gap between the resulting bound and the optimum, expressed as a percentage of the optimum. Table 1 displays, for each set of instances and each procedure, the average gap over the instances in the given set.

As expected, the `K-median` instances are the easiest, and the `Gap-c` instances the most challenging. Also as expected, dual ascent consistently yields worse lower bounds than LP relaxation. As for the four options for producing upper bounds, we performed sign tests to check whether there is a significant ordering between the options. The results can be found in Table 2.

We see that, on average, the best upper bounds are obtained when the dual-based heuristic is applied to an optimal dual solution obtained via the

Table 2: SPLP: Results of sign tests on the quality of the upper bounds.

Hypothesis:	barrier beats ascent	ascent beats cost-based	ascent beats simplex
Result:	$p < 10^{-11}$	$p < 5 \times 10^{-4}$	$p < 5 \times 10^{-9}$

barrier method. Moreover, even when a heuristic dual solution is computed via dual ascent, the upper bounds obtained are typically better than those found by the cost-based heuristic. In our view, this provides strong evidence for the claim that dual information can be useful when driving a primal heuristic.

The big surprise, however, is that the dual-based heuristic performs poorly when the simplex method is used to compute the dual solution. This is so despite the fact that the dual solution is guaranteed to be optimal. A likely explanation for this unusual behaviour is the following. As mentioned in Subsection 2.2, the primal LP is massively degenerate. This means that there are typically a huge number of alternative optimal dual solutions to the LP. Of those, the simplex method will select just one (more or less arbitrary) basic solution, which will be an extreme point of the optimal face in the dual. It seems that the "central" dual solution provided by barrier leads to much more reliable reduced costs than the "extreme" one found by simplex.

To test this hypothesis, we inspected the LP solutions in detail. It turns out that, for most of the instances, over 90% of the basic variables took the value zero in the LP solutions obtained by simplex. Moreover, around 90% of the facility reduced costs were zero, with the remaining 10% being very large. When barrier was used instead, both of these phenomena disappeared.

3.4 Results with randomised procedures

Now we turn our attention to the randomised procedures. Table 3 has a similar interpretation to Table 1, except that each figure is averaged, not only over the instances in the given set, but also over 100 runs of the randomised procedures.

By comparing Table 3 with Table 1, we see that randomisation has little effect on the lower bound obtained with dual ascent. (This is to be expected, since there is nothing special about the order of the clients in the input file.) We also see that perturbing the primal LP causes the lower bound to deteriorate only a little on average, which confirms our belief that the perturbed LP approach yields near-optimal dual vectors. As for the upper bounds, randomisation has no noticeable effect in most cases. When barrier is used, however, the perturbation tends to cause a small worsening. (This too can be confirmed with a sign test.)

Since randomisation leads to multiple primal and dual solutions, a natu-

Table 3: SPLP: Average percentage gaps with randomisation.

		lower bounds		upper bounds			
Set	$m = n$	ascent	LP	ascent	simplex	barrier	cost-based
`Kmed-L`		0.55	0.12	1.19	6.73	0.52	11.0
`Kmed-M`	250	0.29	0.02	0.67	5.97	0.17	7.25
`Kmed-S`		0.00	0.00	0.02	0.04	0.00	0.03
`MO`	100	6.07	3.24	2.47	19.9	0.95	3.56
`MP`	200	7.29	4.65	2.03	22.0	0.60	2.52
`MQ`	300	7.21	4.46	1.91	23.2	0.49	2.20
`KG-S-A`		0.31	0.15	0.13	0.95	0.11	0.48
`KG-S-B`	250	1.47	1.03	0.69	3.38	0.45	1.37
`KG-S-C`		4.31	3.49	0.94	6.65	0.57	2.17
`KG-A-A`		0.30	0.16	0.16	0.91	0.12	0.48
`KG-A-B`	250	1.50	1.05	0.60	3.37	0.37	1.35
`KG-A-C`		4.10	3.33	1.04	7.02	0.62	2.27
`Gap-c`	100	68.8	29.9	42.6	39.9	29.9	41.1

ral strategy is to run the procedures a fixed number of times, and then pick the best lower and upper bounds obtained. The results for this approach can be found in Table 4. We see that this leads to vastly improved upper bounds in almost all cases. Nevertheless, the relative ordering remains much the same, with barrier performing best and dual ascent coming second in most cases.

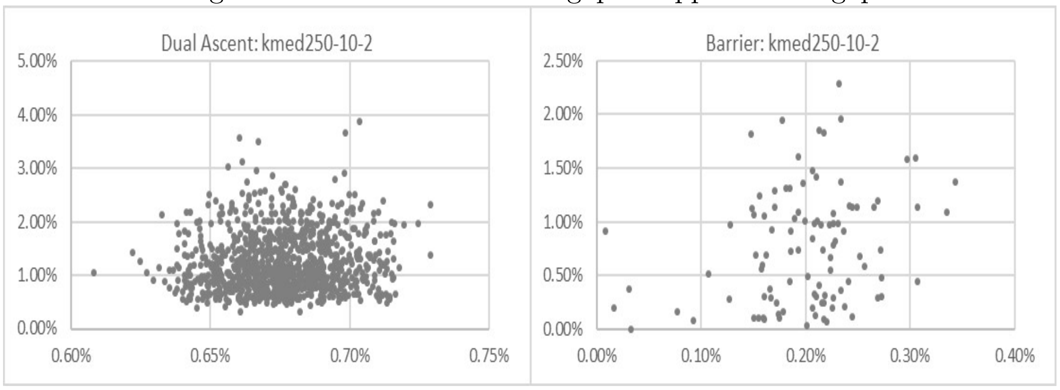
Finally, for each instance and each of three methods (randomised dual ascent, perturbed simplex and perturbed barrier), we produced scatterplots to see whether good lower bounds tended to lead to good upper bounds. Figure 1 shows two such scatterplots for one of the `Kmed-S` instances. The horizontal and vertical axes represent the percentage gap for the lower and upper bound, respectively. The scatterplots indicate that the correlation, if any, is very weak. The same behaviour was found for other instances and methods. In general, we found that the correlation coefficient varied between -0.1 and 0.2, with no discernable pattern.

All things considered, we believe that randomised dual ascent is a very promising approach for large-scale SPLP instances. It is extremely easy to implement and, in our experiments, it was about 100 times faster than the perturbed LP approach. It also has the advantage that one does not need to use LP software.

Table 4: SPLP: Average percentage gaps when randomisation applied and best of 100 bounds taken.

		lower bounds		upper bounds			
Set	$m = n$	ascent	LP	ascent	simplex	barrier	cost-based
Kmed-L		0.49	0.00	0.17	1.71	0.00	7.82
Kmed-M	250	0.13	0.00	0.14	0.82	0.00	6.49
Kmed-S		0.00	0.00	0.00	0.00	0.00	0.03
MO	100	5.93	2.85	0.43	8.29	0.25	0.10
MP	200	7.23	4.20	0.35	10.8	0.12	0.11
MQ	300	7.18	4.04	0.02	12.8	0.23	0.11
KG-S-A		0.29	0.13	0.04	0.66	0.03	0.43
KG-S-B	250	1.45	0.96	0.25	2.01	0.19	0.98
KG-S-C		4.29	3.29	0.10	2.92	0.11	0.75
KG-A-A		0.28	0.14	0.06	0.60	0.05	0.39
KG-A-B	250	1.48	0.98	0.18	2.13	0.10	0.97
KG-A-C		4.09	3.13	0.19	3.04	0.18	0.87
Gap-c	100	56.4	28.2	17.1	21.2	12.9	28.6

Figure 1: SPLP: lower bound gap vs upper bound gap



4 The Set Covering Problem

Now we move on to the SCP. This section is structured in exactly the same way as the previous one.

4.1 Deterministic procedures

To obtain our initial dual solutions for the SCP, we use the same three approaches that we proposed for the SPLP in Subsection 3.1: dual ascent, LP via simplex and LP via interior-point. The only difference is that, for dual ascent, we use the procedure of Balas & Ho [4] described in Subsection 2.3. Our implementation of this procedure runs in $O\big(m(n + \log m)\big)$ time. In practice, it is very fast.

To generate a primal solution from a given dual solution, we use the heuristic of Fisher & Kedia [25], described in Subsection 2.3. Our implementation runs in $O(mn)$ time and is extremely fast in practice.

Again, in order to provide a benchmark, we also considered a primal heuristic that does *not* use dual information. After some experimentation, we settled on Algorithm 2, which is similar to a heuristic in Balas & Ho [4]. With some care, it can be implemented to run in $O(mn)$ time.

Algorithm 2: Greedy Heuristic for Set Covering

```
input  : positive integers $m$, $n$, cost vector $c \in \mathbb{Z}^n_+$.
Set $M := \{1, \ldots, m\}$, $N := \{1, \ldots, n\}$, $C := \emptyset$ and $U := 0$;
repeat
    for all $j \in N$ do
        Let $s(j)$ be the set of rows in $M$ covered by column $j$;
        if $s(j) = \emptyset$ then
            Set $N := N \setminus \{j\}$;
        end
    end
    Let $k \in N$ be the column that minimises $c_j/|s(j)|$;
    Set $M := M \setminus s(k)$, $N := N \setminus \{k\}$, $C := C \cup \{k\}$ and
     $U := U + c_k$;
until $M = \emptyset$;
for all $j \in C$ do
    if $C \setminus \{j\}$ is a cover then
        set $C := C \setminus \{j\}$ and $U := U - c_j$;
    end
end
output: Minimal cover $C$ and associated upper bound $U$.
```

4.2 Randomised procedures

To obtain a large number of good quality dual solutions for the SCP, we use the same two approaches that we used for the SPLP, i.e., *randomised dual ascent* and *perturbed LP*. In each iteration of randomised dual ascent, instead of picking the next row in the sorted list, we pick one row at random from the next five in the list. As for the perturbed LP approach, we simply solve the LP relaxation of (7), but add a small random number to each component of e_m. As in the case of the SPLP, we let these random numbers be uniformly distributed in $[-0.25, 0.25]$.

Finally, we devised a randomised version of Algorithm 2. In each major iteration, instead of selecting the column that minimises $c_j/|s(j)|$, we select a column at random from the five columns with smallest values of $c_j/|s(j)|$. The running time remains $O(mn)$.

4.3 Results with deterministic procedures

As in the case of the SPLP, we implemented all of our lower- and upper-bounding procedures in C. We ran our code on the standard collection of benchmark SCP instances, which are available in the OR-Library [8]. For these instances, the number of variables n is in $\{3000, 4000, 5000, 10000\}$, and the number of constraints m is set to $n/10$. Another parameter is the *density*, which is the expected proportion of non-zero entries in the matrix A. This parameter takes values in $\{2\%, 5\%, 10\%, 20\%\}$. There are 8 sets of instances, labelled 'A' to 'H', with different parameter settings. In each set, there are 5 instances, making 40 instances in total. Data on these instances can be found in the first three columns of Table 5. Optimal values are known for the sets A to F [16]. For sets G and H, we used the best-known upper bounds, again taken from [16].

The last six columns in Table 5 have a similar meaning to the corresponding ones in Table 1. The only difference is that, for sets G and H, the average percentage gaps are with respect to the best-known upper bounds, rather than the optimum.

The results are very different than the ones for the SPLP. In the first place, the lower bounds from dual ascent are of very poor quality, and even the ones from LP relaxation are not great. This is in line with the well-known result that the cost of an optimal SCP solution can be as much as $\ln m$ times larger than the LP bound (see, e.g., [43] and the references therein). As for the upper bounds, there is no evidence that any of the three dual-based heuristics performs better than the cost-based heuristic. In fact, if anything, they look slightly worse. However, the differences are not statistically significant (at the 0.05 level).

Table 5: SCP: average percentage gaps with deterministic procedures

			lower bounds		upper bounds			
Set	n	density	ascent	LP	ascent	simplex	barrier	cost-based
A	3000	2%	25.1	1.51	6.42	6.52	10.5	6.05
B	3000	5%	39.1	7.69	6.93	9.47	10.0	5.07
C	4000	2%	27.8	2.32	6.35	7.27	7.65	5.21
D	4000	5%	41.0	8.27	5.54	9.96	9.19	9.66
E	5000	10%	62.5	24.7	11.9	14.1	10.6	9.99
F	5000	20%	77.1	36.4	15.8	14.5	17.4	15.8
G	10000	2%	47.3	10.1	8.89	8.78	9.51	7.53
H	10000	5%	62.5	23.5	11.9	11.8	11.5	11.9
		Mean	47.8	14.3	9.22	10.3	10.8	8.91

Table 6: SCP: average percentage gaps with randomisation

			lower bounds		upper bounds			
Set	n	density	ascent	LP	ascent	simplex	barrier	cost-based
A	3000	2%	23.6	2.49	5.76	6.72	7.04	10.4
B	3000	5%	39.6	9.26	7.22	8.29	8.49	13.7
C	4000	2%	28.4	3.44	6.49	6.76	6.89	11.7
D	4000	5%	43.3	9.93	7.45	8.96	9.30	14.1
E	5000	10%	63.8	26.9	9.72	11.4	11.3	15.9
F	5000	20%	75.1	38.8	13.0	15.3	15.5	19.3
G	10000	2%	44.5	11.9	8.10	10.2	10.2	10.7
H	10000	5%	62.5	25.5	12.4	13.8	13.6	16.2
		Mean	47.6	16.0	8.74	10.2	10.3	13.9

4.4 Results with randomised procedures

Table 6 presents the gaps obtained with the randomised procedures. By comparing Table 6 with Table 5, we see that randomisation has little effect on the lower bounds. For the upper bounds, however, randomisation causes the cost-based heuristic to perform much worse. Oddly, it also seems to cause the ascent-based heuristic to perform better. Indeed, there is now a clear ordering, with dual ascent coming first and cost-based coming last. This was confirmed by sign tests (Table 7).

Table 7: SCP: Results of sign tests on the quality of the upper bounds.

Hypothesis:	ascent beats simplex	simplex beats barrier	barrier beats cost
Result:	$p < 10^{-6}$	$p < 0.01$	$p < 10^{-12}$

Table 8: SCP: average percentage gaps when randomisation applied and best of 100 bounds taken.

			lower bounds		upper bounds			
Set	n	density	ascent	LP	ascent	simplex	barrier	cost-based
A	3000	2%	17.5	1.51	2.07	2.86	2.77	4.16
B	3000	5%	30.5	7.69	0.79	0.79	1.59	3.72
C	4000	2%	21.4	2.32	3.00	2.90	3.08	4.52
D	4000	5%	34.2	8.27	2.13	3.44	2.86	4.35
E	5000	10%	52.1	24.7	1.41	2.07	2.07	3.55
F	5000	20%	65.8	36.4	4.40	4.40	4.40	7.16
G	10000	2%	37.8	10.1	4.09	5.77	5.77	6.50
H	10000	5%	55.2	23.5	7.42	7.39	6.93	10.2
		Mean	39.3	14.3	3.16	3.70	3.68	5.52

Table 8 shows the results obtained when randomisation is applied 100 times and the best bounds are retained. As in the case of the SPLP, this leads to vastly improved upper bounds in almost all cases. Nevertheless, the relative ordering remains much the same, with dual ascent performing best and cost-based coming last in most cases. This confirms our belief that dual information, if used intelligently, can be useful for guiding primal heuristics. It also makes us more convinced that randomised dual ascent is a promising technique for producing upper bounds quickly.

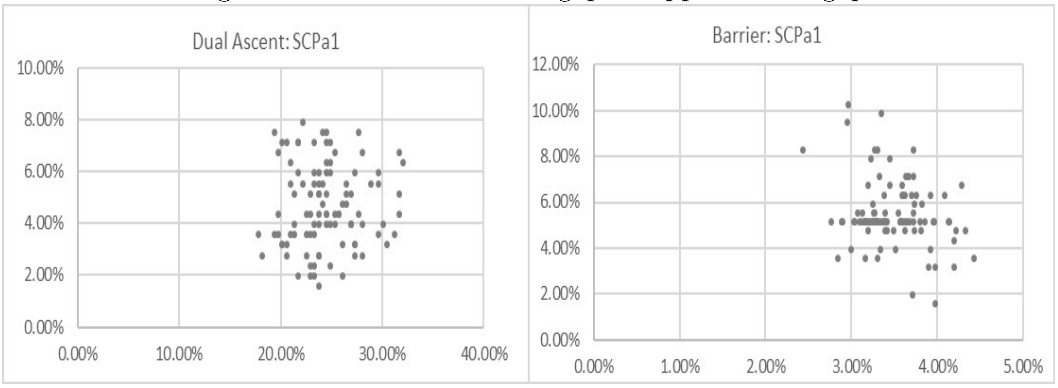
Observe that, in the case of the SCP, the choice between simplex and barrier has little effect on the quality of the resulting upper bounds. This is very different from what we saw for the SPLP. The reason is probably that primal degeneracy is less of an issue for the SCP. To confirm this, we check the LP solutions in detail. In the solutions obtained by the simplex method, only around 10% of the basic variables were at zero, compared with around 90% in the case of the SPLP.

Finally, Figure 2 shows two typical scatterplots obtained for the SCP. As in the case of the SPLP, there is no obvious positive correlation between the quality of the lower and upper bounds. In fact, the scatterplot on the right suggests a small negative correlation. In general, we found that the correlation coefficient varied between -0.1 and 0.1, with no discernable pattern.

5 Conclusion

At the time of writing, dual-based heuristics have been around for about fifty years. Although more sophisticated heuristics (and indeed meta-heuristics) exist, dual-based heuristics are easy to code, scale well with problem size, and tend to give solutions of acceptable quality in practice. We have shown,

Figure 2: SCP: lower bound gap vs upper bound gap



however, that they can behave in counter-intuitive ways. In particular, if the primal LP is highly degenerate, then a simple greedy dual heuristic may lead to better upper bounds than those obtained when the dual is solved to optimality via the simplex method.

The main lesson from our work is that, when faced with a given combinatorial optimisation problem, it is well worth coding and testing several different heuristics for finding dual solutions, before jumping to conclusions about the effectiveness of dual-based heuristics. We believe that it is worth coding and testing more than one primal heuristic as well.

References

[1] S. Ahn, C. Cooper, G. Cornuéjols & A.M. Frieze (1988) Probabilistic analysis of a relaxation for the p-median problem. *Math. Oper. Res.*, 13, 1–31.

[2] A. Balakrishnan, T.L. Magnanti & R.T. Wong (1989) A dual-ascent procedure for large-scale uncapacitated network design. *Oper. Res.*, 37, 716–740.

[3] E. Balas & M.C. Carrera (1996) A dynamic subgradient-based branch-and-bound procedure for set covering. *Oper. Res.*, 44, 875–890.

[4] E. Balas & A. Ho (1980) Set covering algorithms using cutting planes, heuristics, and subgradient optimization: a computational study. *Math. Program. Study*, 12, 37–60.

[5] M. Balinski (1965) Integer programming: methods, uses, computation. *Mgnt. Sci.*, 12, 254–313.

[6] J.E. Beasley (1987) An algorithm for set covering problems. *Eur. J. Oper. Res.*, 31, 85–93.

[7] J.E. Beasley (1990) A Lagrangian heuristic for set covering problems. *Nav. Res. Log.*, 37, 151–164.

[8] J.E. Beasley (1990) OR-Library: distributing test problems by electronic mail. *J. Oper. Res. Soc.*, 41, 1069–1072.

[9] J.E. Beasley (1993) Lagrangian heuristics for location problems. *Eur. J. Oper. Res.*, 65, 383–399.

[10] J.E. Beasley & K. Jörnsten (1992) Enhancing an algorithm for set covering problems. *Eur. J. Oper. Res.*, 58, 293–300.

[11] C. Beltran-Royo, J.-P. Vial & A. Alonso-Ayuso (2012) Semi-Lagrangian relaxation applied to the uncapacitated facility location problem. *Comput. Optim. Appl.*, 51, 387–409.

[12] O. Bilde & J. Krarup (1977) Sharp lower bounds and efficient algorithms for the simple plant location problem. *Ann. Discr. Math.*, 1, 79–88.

[13] M.A. Boschetti, A. Mingozzi & S. Ricciardelli (2008) A dual ascent procedure for the set partitioning problem. *Discr. Optim.*, 5, 735–747.

[14] E.K. Burke & G. Kendall (eds.) (2014) *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques* (2nd edn.). Springer US.

[15] A. Caprara, M. Fischetti & P. Toth (1999) A heuristic method for the set covering problem. *Oper. Res.*, 47, 730–743.

[16] A. Caprara, M. Fischetti & P. Toth (2000) Algorithms for the set covering problem. *Ann. Oper. Res.*, 98, 353–371.

[17] S. Ceria, P. Nobili & A. Sassano (1998) A Lagrangian-based heuristic for large-scale set covering problems. *Math. Program.*, 81, 215–228.

[18] M. Conforti, G. Cornuéjols & G. Zambelli (2014) *Integer Programming.* Cham, Switzerland: Springer.

[19] L.. Drummond, M. Santos & E. Uchoa (2009) A distributed dual ascent algorithm for Steiner problems in multicast routing. *Networks*, 53, 170–183.

[20] P.S. Efraimidis & P.G. Spirakis (2006) Weighted random sampling with a reservoir. *Infor. Proc. Lett.*, 97, 181–185.

[21] D. Erlenkotter (1978) A dual-based procedure for uncapacitated facility location. *Oper. Res.*, 26, 992–1009.

[22] E. Feldman, F.A. Lehrer & T.L. Ray (1966) Warehouse location under continuous economies of scale. *Mgmt. Sci.*, 12, 670–684.

[23] M. Fischetti, I. Ljubić, M. Sinnl (2017) Redesigning Benders decomposition for large-scale facility location. *Mgmt. Sci.*, 63, 2146–2162.

[24] M.L. Fisher (1981) The Lagrangian relaxation method for solving integer programming problems. *Mgmt. Sci.*, 27, 1–18.

[25] M.L. Fisher & P. Kedia (1990) Optimal solution of set covering/partitioning problems using dual heuristics. *Mgmt. Sci.*, 36, 674–688.

[26] R. Galvão & L. Raggi (1989) A method for solving to optimality uncapacitated location problems. *Ann. Oper. Res.*, 18, 225–244.

[27] A.M. Geoffrion (1974) Lagrangean relaxation for integer programming. *Math. Program. Study*, 2, 82–114.

[28] D. Ghosh (2003) Neighborhood search heuristics for the uncapacitated facility location problem. *Eur. J. Oper. Res.*, 150, 150–162.

[29] M. Guignard (1988) A Lagrangian dual ascent algorithm for simple plant location problems. *Eur. J. Oper. Res.*, 35, 193–200.

[30] M. Guignard (2003) Lagrangean relaxation. *Trabajos de Operativa (TOP)*, 11, 151–228.

[31] S. Haddadi (1997) Simple Lagrangian heuristic for the set covering problem. *Eur. J. Oper. Res.*, 97, 200–204.

[32] P. Hansen, J. Brimberg, D. Urošević & N. Mladenovič (2007) Primal-dual variable neighborhood search for the simple plant-location problem. *INFORMS J. Comput.*, 19, 552–564.

[33] J. Janáček & L. Buzna (2008) An acceleration of Erlenkotter–Körkel's algorithms for the uncapacitated facility location problem. *Ann. Oper. Res.*, 164, 97–109.

[34] M. Jünger *et al.* (eds.) (2009) *50 Years of Integer Programming: 1958-2008.* Berlin: Springer.

[35] Y. Kochetov & D. Ivanenko (2005) Computationally difficult instances for the uncapacitated facility location problem. In T. Ibaraki, K. Nonobe & M. Yagiura (eds) *Metaheuristics: Progress as Real Problem Solvers*, pp. 351–367. Boston, MA: Springer.

[36] M. Körkel (1989) On the exact solution of large-scale simple plant location problems. *Eur. J. Opl Res.*, 39, 157–173.

[37] J. Kratica, D. Tosic, V. Filipovic & I. Ljubic (2001) Solving the simple plant location problem by genetic algorithm. *RAIRO Oper. Res.*, 35, 127–142.

[38] A.N. Letchford & S.J. Miller (2012) Fast bounding procedures for large instances of the simple plant location problem. *Comput. Oper. Res.*, 39, 985–990.

[39] V. Maniezzo, T. Stützle & S. Voss (eds.) (2009) *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming.* Heidelberg: Springer.

[40] J.G. Morris (1978) On the extent to which certain fixed-charge depot location problems can be solved by LP. *J. Oper. Res. Soc*, 29, 71–76.

[41] T. Polzin & S.V. Daneshmand (2001) Improved algorithms for the Steiner problem in networks. *Discr. Appl. Math.*, 112, 263–300.

[42] M. Posta, J.A. Ferland & P. Michelon (2014) An exact cooperative method for the uncapacitated facility location problem. *Math. Program. Comput.*, 6, 199–231.

[43] R. Saket & M. Sviridenko (2012) New and improved bounds for the minimum set cover problem. In A. Gupta *et al.* (eds.), *Proc. APPROX-RANDOM XV*, pp. 288–300. Berlin: Springer.

[44] M.J. Todd (1982) An implementation of the simplex method for linear programming problems with variable upper bounds. *Math. Program.*, 23, 34–49.

[45] D.P. Williamson (2002) The primal-dual method for approximation algorithms. *Math. Program.*, 91, 447–478.

[46] R.T. Wong (1984) A dual ascent approach for Steiner tree problems on a directed graph. *Math. Program.*, 28, 271–287.

[47] B. Yelbay, Ş.İ. Birbil & K. Bülbül (2015) The set covering problem revisited: an empirical study of the value of dual information. *J. Ind. Mgmt. Optim.*, 11, 575–594.