Transmission Tower Painting Optimisation

C. McLean, A. Downward, G. Zakeri, A. Philpott
Department of Engineering Science, University of Auckland
cmcl076@auckland.ac.nz

Abstract

Transpower manages nearly 25,000 galvanised steel lattice towers, and requires a maintenance strategy to prevent tower corrosion. Each tower is rated with a condition assessment: the deterioration of which can be predicted by its corrosion zone, defined by the climate and location. When towers are in close proximity, they can be painted concurrently, thus reducing the total time to paint.

A modified Bellman-Ford algorithm was developed to generate sets of feasible crew painting schedules. The best schedules were then chosen using two different approaches: a greedy selection method and a Set Partitioning problem. The greedy approach involved sequentially selecting the best schedule generated by the modified Bellman-Ford algorithm and removing these towers from the network. This performed as expected; only the most urgent towers in the network were covered. The Set Partitioning problem delivered a set of viable schedules that covered urgent towers, as well as less urgent towers when in close proximity. In doing so, the schedules were able to make the most of the ability to paint towers concurrently.

Key words: Set Partitioning, Bellman-Ford, scheduling.

1 Introduction

Transpower controls the national electricity grid. They own and operate around 41,000 transmission line support structures. Nearly 24,800 of these structures are galvanised steel lattice towers, otherwise known as pylons. Over the course of their operational lives, they face deterioration from a number of factors and, as such, require periodic maintenance to prevent tower failure. The preferred maintenance option involves conducting tower painting to protect the steel from corrosion. However, if the corrosion is too great, a full tower replacement may be required.

Transpower is looking to implement an optimal schedule in order to manage tower corrosion. An ideal set of schedules produced by this optimisation problem need not cover only the urgent towers. To be best placed for future maintenance, towers on the same line should be maintained such that they end up with similar conditions. As a result, later maintenance will only require crews to repaint the towers line by line.

1.1 Tower Network and Painting

The country has been divided up into three different regions. Given the large number of towers in the network, the study was limited to one region, as a proving ground for the analytical methods. Transpower identified the southern North Island, with five contractor crews and 7050 towers, as having the highest priority.

On average, a crew will take about seven to nine days to paint a single tower. Crews will typically work on a line of towers. If this is the case, crews can then work on a number of towers concurrently. For this study, we have modelled such a case as taking four days per individual tower.

1.2 Tower Corrosion

The main issue that Transpower faces in regards to its transmission towers is the corrosion of the steel. To combat this, teams conduct periodic condition assessments (CA) on towers in order to determine the current state of the structure. These are performed every eight to ten years, depending on the age of the structure.

Each CA rating is between 0 and 100, where 100 denotes a new tower in perfect condition. Typically, towers with an average CA rating less than 20 must be replaced as this is considered the end of life of the structure.

1.2.1 Corrosion Zones

Every tower has been assigned one of six zones, defined by the local climate and geographical features. These were defined by existing Transpower tower data.

For example, extreme zones are restricted to coastal areas near Bluff and New Plymouth, as well as some geothermal areas around the Volcanic Plateau. Coastal regions in Northland and Taranaki are considered either very severe or severe. At the other end of the scale, benign regions are found around Central Otago.

The corrosion zones are used to determine a number of aspects related to tower condition assessments, including the deterioration rate of the tower condition and the optimal condition for tower repainting.

Corrosion curves are used to model the rate of deterioration of unpainted towers in each corrosion zone. Transpower's existing corrosion bands, found in Transpower (2013), were approximated using second-order polynomials to give Figure 1.

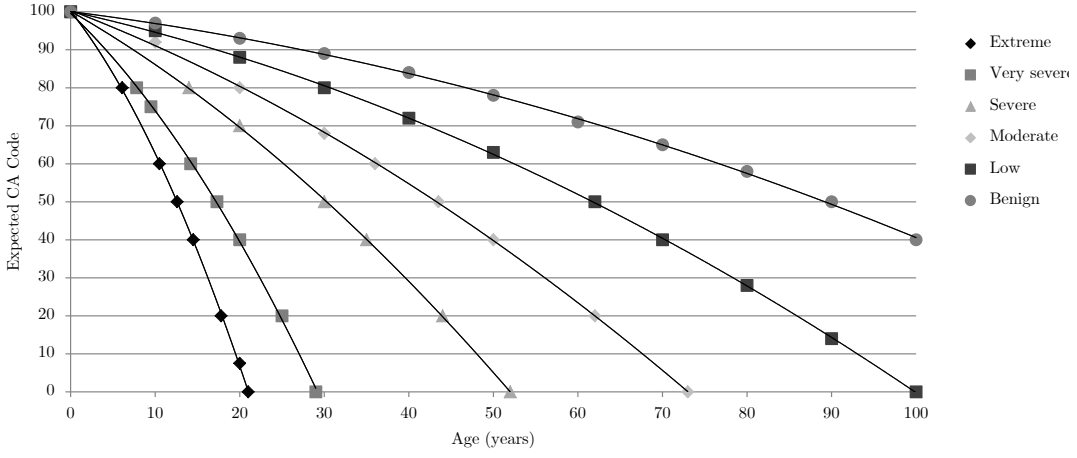


Figure 1: Quadratic fit to unpainted galvanised steel tower degradation curves.

2 Cost Prediction

To simplify the model, we only considered the outturn painting costs provided by Transpower. These were based on the current condition of the tower. All other costs, such as travel or land access costs, were either considered constant, or are managed by the contractor, hence were irrelevant in this study.

2.1 Optimal Cost of Painting

The model minimises the cost of not painting towers in this current period, that is, the cost of deferring painting until some later date. This accounts for the urgency required when painting certain towers, while not naively attempting to paint all the towers in the same period.

Some towers will have a negative cost of deferral: these towers will be cheaper to paint at a later date. In this study, we want to paint as many towers with a positive cost of deferral as possible as it will be more expensive to paint these towers later.

To find the optimal cost of not painting a tower, it is necessary to compare the cost of painting the tower in the current year with the cost of painting the tower in its "optimal" painting year. The optimal year to paint a tower is defined as the year where the net present cost of painting the tower is at a minimum compared to the cost over other years. This represents the balance between the value of money and the cost of painting the tower in that particular period.

The deferral cost is the difference between the cost of painting a tower at its optimal condition and the cost of painting its current condition.

3 Adjacent Towers

In order to define which structures could be painted concurrently, we needed to capture the tower adjacency in the transmission network. Tower a is adjacent to tower b if painting work can be performed simultaneously on both a and b.

The method used defined at least three towers adjacent to every tower. It was set up so that towers found in high density areas would still be considered adjacent to many other towers. Therefore, our adjacency network reliably captured the complexity of the actual transmission network in a simple manner.

These adjacencies are used in the modified Bellman-Ford algorithm (Section 4.1) when determining the painting time for a sequence of towers. (Recall that the set-up time is reduced when adjacent towers are painted concurrently.)

4 Model

4.1 Initial Approach

Initially, we aimed to model the problem by generating the set of schedules, either by complete enumeration or some form of column generation, and optimise it as a Vehicle Routing problem. The length of the tour would define the schedules' costs; a Travelling Salesperson problem (TSP) would be solved to find the minimum distance travelled by the crew for each schedule. However, we learned that the current process involved assigning a set of towers to a contractor, who would then decide the scheduling within this themselves. As such, the TSP costs were irrelevant.

4.2 Set Partitioning Problem

The master problem is formulated as a Set Partitioning problem (SPP).

Indices

t = towers in region: $T = 1, \ldots, m$;

Parameters

y = number of crews scheduled to work in a particular region;
n = number of possible work schedules to which a crew can be assigned;
c_t = cost of deferring the painting of a tower t, such that $c = [c_1, c_2, ..., c_m, 0, ..., 0]$;
I = identity matrix capturing deferral of painting the tower until next season;
M = set of possible schedules that crews can undertake,

$$M_{m\times n} : (M)_{tj} = \begin{cases} 1 & \text{if tower } t \text{ is painted in work schedule } j, \\ 0 & \text{otherwise;} \end{cases}$$

G = GUB constraint, where

$$G_{m+n} : G_j = \begin{cases} 0 & 1 \leq j \leq m \\ 1 & (m+1) \leq j \leq (m+n). \end{cases}$$

Model SPP

$$\begin{aligned} \min \quad & c^T x \\ \text{s/t} \quad & \left[I_{m\times m} \vdots M \right] x \quad = \mathbf{e} \quad [\pi_i] \qquad (1) \\ & Gx \qquad\qquad \leq y \quad [\mu] \qquad (2) \\ & x \in \{0,1\}^{m+n}. \end{aligned}$$

Explanation The objective is to minimise the cost of deferring the painting of all towers $t \in T$. It is desirable that towers with a positive c_i, (cost of deferral), be covered in one of the n schedules. Towers that are cheaper to paint at a later date, with a negative cost of deferral, can either be painted in a schedule or deferred as required, to produce an optimal solution.

A small version of the problem, using artificial towers and costs, was set up as a Set Partitioning problem and solved using OpenSolver in Excel. The model and its solution are shown in Table 1.

Table 1: Excerpt of small SPP model and solution in Excel.

	x_1	x_2	x_3	...	x_{14}	x_{15}	x_{16}	x_{17}	x_{18}	x_{19}	x_{20}	x_{21}		***Objective:***	-35
min	0	0	0	...	0	0	40	-20	-15	-10	10	5			
vars	0	0	1	...	0	1	0	1	1	0	0	0			
s.t															*RHS*
crew	1	1	1	...	1	1	0	0	0	0	0	0	=	2	2
T 1	1	1	1		0	0	1	0	0	0	0	0	=	1	1
T 2	1	0	0	...	0	0	0	1	0	0	0	0	=	1	1
T 3	0	1	0		0	0	0	0	1	0	0	0	=	1	1
T 4	0	0	1		1	0	0	0	0	1	0	0	=	1	1
T 5	0	0	0	...	0	1	0	0	0	0	1	0	=	1	1
T 6	0	0	0		1	1	0	0	0	0	0	1	=	1	1

Once it was established that this produced viable results, the problem could then be set up in Python, using Python for Gurobi – a faster optimisation solver.

4.3 Modified Bellman-Ford Algorithm

Due to the large number of potential schedules, we implemented a column-generation method, as described in Barnhart et al. (1998) and O'Sullivan et al. (2012). This would provide a list of suitable schedules to try in the SPP constraint matrix.

We implemented a bi-objective, constrained version of the Bellman Ford algorithm to generate columns. The algorithm aims to minimise both the cost of painting the towers, as well as the time taken to paint them. It is constrained by a time period, ensuring that all painting schedules can be completed in the time available. This algorithm returns the set of possible schedules that a crew could perform.

The standard Bellman-Ford algorithm generates the shortest paths from a single source, or "crew", node to all other nodes in a weighted digraph, from Leiserson et al. (2001). We have modified this algorithm in order to generate columns for the Set Partitioning problem.

The network is set up as a representation of the towers in a particular region:

- weighted edges represent the time taken to paint a tower, conditioned on the previous tower;
- "prizes" at the nodes are the duals computed in the Set Partitioning problem.

Figure 2 shows the network configuration. Adjacent edges, defined in Section 3, represent towers that can be painted concurrently. These edges will have weight $w_a = 4$ days.

To represent the time taken to paint two towers that crews cannot work on concurrently, we will use a dummy node. The dummy node eliminates the need to have an edge between every node. This reduces the complexity of the network. The weight of these non-adjacent edges will be defined by $w_{na} = 9$ days.

The algorithm generates a set of efficient solutions for the best total prize and duration of a schedule of towers from the "crew" or "source" node to any other node in the network. The schedules must be able to be completed within the set time period. They can then be used as columns in the Set Partitioning problem.

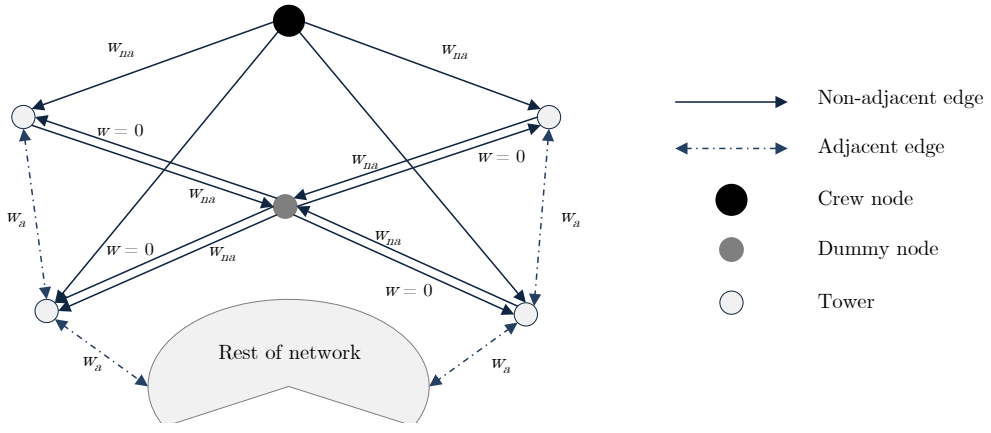


Figure 2: Representation of modified Bellman-Ford network.

Negative weight cycles are possible in this network, since some prizes at the nodes will have a negative value. However, these are avoided by enforcing the condition that a path cannot visit a node more than once. This correlates well with our tower network as it means that a tower cannot be painted more than once in a schedule.

4.3.1 Representation of Network

We will be using a directed graph to represent the network of towers.

v_t be the vertex representing tower t for any $t \in T$;

S_1 be the source/crew node;

S_2 be the dummy node.

Then let V denote the set of vertices in the directed network as follows

$$V = \{v_t : t \in T\} \cup \{S_1, S_2\}$$

Let A be the pre-existing adjacency matrix for the network, defined in Section 3, where $a_{ij} = 1$ if towers i and j are adjacent and 0 otherwise. Let:

E_1 be the set of all edges defined by the adjacency matrix:

$$E_1 = \{(v_i, v_j) : a_{ij} = 1, i, j \in T\}, \text{with weight } w_i = w_a \text{ for } i \in E_1;$$

E_2 be the set of edges from the source node to every node in towers:

$$E_2 = \{(S_1, v_j) : j \in T\}, \text{with weight } w_i = w_{na} \text{ for } i \in E_2;$$

E_3 be the set of edges from the dummy node to every node in towers:

$$E_3 = \{(S_2, v_j) : j \in T\}, \text{with weight } w_i = 0 \text{ for } i \in E_3;$$

E_4 be the set of edges from every node in towers to the dummy node:

$$E_4 = \{(v_i, S_2) : i \in T\}, \text{with weight } w_i = w_{na} \text{ for } i \in E_4.$$

Then let E denote the set of edges in the network as follows

$$E = E_1 \cup E_2 \cup E_3 \cup E_4.$$

Step 1: This step is used to set up the network by initialising node prizes with the dual variables from the SPP and setting time totals to infinity (typical of Bellman-Ford implementations).

Each node will have a list of labels that define the set of efficient paths from the source node to that particular node. These labels keep record of the total prize of that particular path, the total time taken from the source node to that node and the list of predecessor nodes.

Step 2: This is the main algorithm in the modified Bellman-Ford procedure. It is similar to the single objective Bellman-Ford algorithm except that in order to account for the bi-objective nature of our algorithm, it includes a dominance check that will store only efficient and weakly efficient paths to each node.

Algorithm 1 Relax edges repeatedly

```
timeLimit ← maximum time period
  for all i = 1 to (|V| − 1) do
    changed = False
    for all e in E do
      u ← e.StartNode
      v ← e.EndNode
      for all l in u.labels do
        candidateTime ← l.totalTime + e.weight
        if candidateTime ≤ timeLimit then
          if v is dummy OR v not in l.predecessors then
            Check dominance of new candidate label
            if l is efficient or weakly efficient then
              changed ← True
    if changed is False then
      BREAK
```

Step 3: Labels, or non-dominated points, are sorted numerically according to prize and time. The algorithm compares the prize and time values of the candidate label with existing labels to determine if it is dominated. Dominated labels are removed from the node.

Output: The modified Bellman-Ford algorithm returns a set of possible schedules that can be used as columns in the Set Partitioning Problem. These schedules define the towers that a crew should paint within the given time period, given the current solution to the master problem.

4.4 Reduced Cost Analysis

In order to generate new columns for our master problem, we need to consider the reduced costs. We want to generate columns with negative reduced costs in order to improve the current solution. Hence at optimality, the reduced costs in this study must be non-negative.

4.5 Assembling the Algorithms

The best p unique schedules generated by the modified Bellman-Ford algorithm are used as entering columns in the relaxed Set Partitioning problem. This then returns the shadow prices to be used as costs in the next iteration of the modified Bellman-Ford algorithm. The algorithms iterate until all reduced costs are non-negative or an iteration limit is met. The final iteration solves the integer program (IP).

As the problem may run out of memory, we can either change the bound gap, or remove schedules from the problem. We then rerun the optimisation.

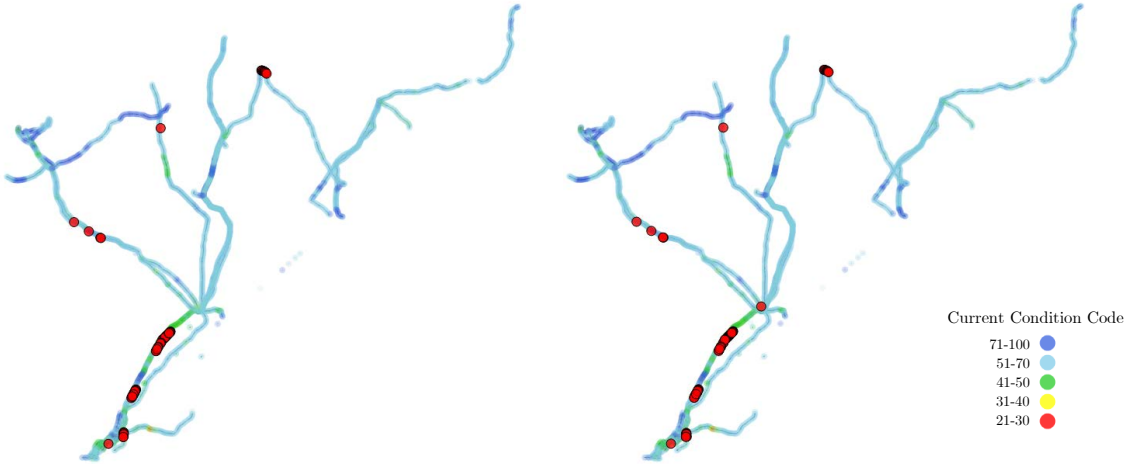
5 Results

The algorithms were run on the southern North Island network, covering 7050 towers, with five painting crews assigned to the region. A schedule period of 90 days was set, as this allowed the problem to be solved in a timely fashion.

5.1 Greedy Heuristic

The greedy heuristic involved running the modified Bellman-Ford algorithm, keeping the best schedule it produced and then removing the towers in that schedule from the network. The algorithm would then be re-run until the required number of schedules were produced: one for each crew in the region. These schedules paint the towers in the most urgent conditions. Some towers are covered individually, as opposed to being painted with other towers on the same line, as seen in Figure 3a. These towers would need to have had a high cost of deferral in order to have overcome the penalty of moving equipment and setting up again (denoted by the edge weight in the network).

This method covers the towers with the poorer conditions, specifically most of those towers with a condition code less than 50. The greedy heuristic readily paints non-adjacent towers where necessary; the benefits from adjacent towers being painted concurrently are not utilised.



(a) Greedy approach (b) SPP approach

Figure 3: Towers painted by 5 crews in 90 days, southern North Island.

5.2 Set Partitioning Model

Running the code with the Set Partitioning model produced a variety of results. Using $p = 1000$ entering columns introduced memory problems when trying to solve the integer program: the full IP could not be solved. $p = 100$ and $p = 200$ produced more manageable problem sizes that could be run without relying on heuristic methods to yield a viable solution.

After 25 iterations, using $p = 1000$ entering columns, the integer program produced five viable painting schedules, in Figure 3b. Running the Set Partitioning model for a further 25 iterations, while also producing five schedules, covered even fewer towers. Table 2 shows the distribution of the number of towers per schedule and the total cost savings made by painting those towers in the current period.

The best results for the Set Partitioning problem were found when we used $p = 200$ entering columns for each iteration. The starred column in Table 2 shows the results from this run of the SPP. The distribution of towers to be painted in this period shows a tendency for a crew to cover one particular sub-region. Crews may paint a couple of towers in each of the worse sub-regions, where many urgent towers are found. This set of schedules painted the same number of towers as in the greedy

case, yet produced the greatest cost savings while doing so. This choice of $p = 200$ also meant that the SPP was small enough to be solved easily in Gurobi, without having to introduce any bound gaps on the objective value, or facing memory issues.

Unlike the greedy heuristic, the Set Partitioning problem is more likely to paint towers in series, along the same line. There are fewer "jumps" to non-adjacent, lower condition towers than are evident with the greedy approach.

Table 2: Number of towers painted in each schedule over a 90 day period (number of iterations).

	Greedy	**SPP (25) 1000 entering cols.**	**SPP (50) 1000 entering cols.**	**SPP (50)* 200 cols.**	**SPP (50) 100 cols.**
Schedule 1	20	20	16	20	20
Schedule 2	17	16	16	15	17
Schedule 3	15	13	14	16	11
Schedule 4	11	15	14	15	16
Schedule 5	18	14	16	15	15
Num. towers	81	78	76	81	79
Savings	522,696	508,384	494,304	524,201	502,172

6 Discussion

Choosing the most suitable number of entering columns in each iteration (p) is a balancing act. If p is too small, the SPP does not gain a wide variety of the possible columns generated by the modified Bellman-Ford algorithm. Conversely, a large p will satisfactorily add a wider variety of possible schedules to the SPP. However, since no columns are ever removed from the basis during the LP iterations, the number of binary variables soon becomes unmanageable and the Gurobi solver has insufficient memory to solve the IP.

The greedy heuristic paints the towers with the worst conditions, while constrained by a time limit. By removing the towers in the best schedule from the next iteration, it eliminates the ability for the model to select the less urgent towers. In doing so, it removes the possibility of arriving at the desired outcome in Section 1, where future maintenance occurs ideally on a line-by-line basis.

In terms of objective value, the SPP obtained the best objective when run with $p = 200$ entering columns, seen in Table 2. This illustrates that the SPP produced a solution that would create the greatest cost savings for this current period, while attempting to cover towers in series.

Many towers in the region have similar condition assessments and are in the same corrosion zones, thus the predicted optimal condition for painting will be the same. As a result, numerous towers will have the same cost of painting this period. Many of the schedules generated by the modified Bellman-Ford algorithm will have the same predecessor towers because of these comparable tower prizes introducing a degree of symmetry into this shortest path problem. Hence the Set Partitioning problem will have issues finding some suitable, viable schedule that satisfies all constraints.

This model currently produces a static set of schedules for contractors to use. Calculating the optimal condition for painting, as in Section 2.1, as opposed to simply calculating the cost of painting in the next period, introduces a sense of continuity to the model. By using this minimum cost, it allows the model to capture the importance and benefit of painting a tower when it is cheapest to do so.

7 Conclusions

A Set Partitioning model, with a modified Bellman-Ford algorithm as a column-generation method, was implemented to solve the optimisation problem involving the painting maintenance strategy for Transpower's transmission towers.

A greedy heuristic was used to generate a first-cut approximation of the schedules. The schedules generated by this approach tended to cover only the poorer towers in the network.

The Set Partitioning problem and modified Bellman-Ford column generation were run sequentially for a set number of iterations, using several numbers of entering columns per iteration. When run for a period of 90 days, the SPP model generated five schedules: one for each of the crews. The best SPP solution used only 200 entering columns per iteration and produced favourable results compared to the greedy approach. The schedules generated by the Set Partitioning model tended to paint towers in series. Adjacent towers in a less urgent condition could be scheduled in order to paint more urgent towers in a timely manner.

7.1 Future Work

The next stage of this optimisation problem would be to incorporate a more dynamic programming approach. This should be able to account for the painting of towers whose optimal condition has yet to be reached. This could either be by solving for multiple time periods, or changing the cost of not painting into a cost to go function.

The model could be extended to cover the entire country. We could assess the value of adding additional crews, or transferring crews from one region to another.

A method can be developed for removing columns from the SPP, to ensure that the problem is tractable and without compromising the integer solution.

References

Barnhart, C., E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance. 1998. "Branch-and-price: Column generation for solving huge integer programs." *Operations research* 46 (3): 316–329.

Leiserson, C.E., R.L. Rivest, C. Stein, and T.H. Cormen. 2001. *Introduction to algorithms*. The MIT press.

O'Sullivan, M, Q Lim, C Walker, I Dunning, and S Mitchell. 2012. *Dippy: A Simplified Interface for Advanced Mixed-integer Programming*. Department of Engineering Science, School of Engineering, University of Auckland.

Transpower. 2013. "Extract from Draft Fleet Strategy for Tower Painting." Technical Report, Transpower.