# Optimal Scheduling of Earth-Imaging Satellites with Human Collaboration via Directed Acyclic Graphs

**Sean Augenstein**
Skybox Imaging
1061 Terra Bella Ave
Mountain View, California 94043

**Abstract**

Optimal scheduling of Earth-observing satellites is crucial to satisfying Skybox Imaging's customers with timely high-resolution imagery and HD video. The company has implemented software that automatically plans which ground targets are imaged by its satellites, maximizing overall utility while not exceeding physical limitations. The software is responsive to user interactions, e.g., additions of new targets or explicit forcing of an existing target (into or out of the schedule). The result is a real-time collaboration between autonomous scheduler and human collection manager, with the former aware of how to optimize each satellite's image collection and the latter aware of late-breaking changes affecting target desirability. The scheduler encodes the problem as directed acyclic graphs (DAGs): nodes represent imaging opportunities and edges (or lack thereof) encode agility performance of the satellite. The optimal schedule of targets is the highest weighted path through a DAG. Human actions map to addition/subtraction of edges in the DAG. This paper discusses the graph-based optimization around these human interactions, and some properties of the problem that allow for computational savings.

## Introduction

Skybox Imaging (Skybox) provides easy access to reliable, frequent high-resolution imagery and HD video of the Earth by combining the power of web technologies and the world's first coordinated satellite constellation. Each satellite complete a revolution of the Earth once every 98 minutes, collecting imagery over desired targets and downlinking the data during ground station passes. With a constellation of such satellites, Skybox can provide customers with an image of anywhere on the Earth at least once a day, enabling a new frontier of big data applications built around timely geo-spatial information.

As the company's business model is heavily leveraged on efficient utilization of its satellites, a sound approach to schedule optimization is critical to satisfying customers' demands for high volumes of Earth imagery. A solution needed to be capable of rapid adaption and recalculation, allowing for "forcing in" of top priority targets (e.g., breaking news events), and "forcing out" of temporarily undesirable targets (e.g., targets obscured by clouds).

Figure 1: Two “SkySats” Awaiting Launch

This paper describes the optimization algorithm developed at Skybox over the last 3 years. The algorithm enables human-computer interaction, and is inspired by a classic problem of computer science, that of finding the shortest path through a graph (Dijkstra 1959) (Bellman 1956) (Ford and Fulkerson 1962). As opposed to shortest path, Skybox's algorithm calculates highest weighted path, to select a maximal feasible set of events to schedule.

The specifics of the Skybox problem (a mixed discrete/continuous optimization problem) are described, and a short background on previous research in satellite scheduling and path problems is given. This is followed by the solution developed at Skybox for performing this optimization, namely, fully discretizing the problem and constructing directed acyclic graphs (DAGs) with edges encoding the satellite's agility constraints. DAGs are desirable as they allow for quick schedule recalculation after human collaboration induces incremental changes to the edges of the DAG. The specifics of forcing out, forcing in, and determining feasibility of future force ins are explained. Some computational savings due to the transitive properties of the graph are also discussed.

## The Problem

Let a set of pointing *windows* exist, where each pointing window $i$ has an importance weight $w_i$, a desired event duration $\tau_i$, an earliest possible start time $st_i$, and a latest possible end time $et_i$. The windows are known, as they are determined from orbital dynamics and the geographic locations of places on the Earth where the satellite should collect an image.

$$
\begin{aligned}
&\max_{\bar{c}} \sum_{i=0}^{N} c_i w_i \\
&\textbf{s.t. } (\forall\, i) \ : \\
&\qquad c_i \in 0, 1 \\
&\qquad t_i > st_i \\
&\qquad t_i + \tau_i < et_i \\
&\qquad t_j - (t_i + \tau_i) \geq c_j \times c_i \times f_{agility}\left(\theta_i(t_i + \tau_i), \theta_j(t_j)\right)
\end{aligned}
$$

The objective of the satellite schedule optimization problem is to select a set of pointing *events* (where each event falls within a corresponding pointing window) to achieve maximum sum over their importance weights. A pointing event represents the act of pointing at a ground target over an exact time range. The variable $c_i$ indicates whether a pointing event from pointing window $i$ has been selected or not ($\bar{c}$ is the overall selection vector, the output of the optimization).

Each pointing event starts at a time $t_i$ so that $t_i$ and $t_i + \tau_i$ are within the continuous range $[st_i, et_i]$ of the corresponding pointing window, and such that the satellite can feasibly change its orientation $\theta(t)$ from one pointing event $i$ to the next (in chronological order) pointing event $i + 1$.

An agility function $f_{agility}()$ takes two orientations and returns the time necessary to change the satellite's pointing from one orientation to the other. The agility function $f_{agility}()$ is known; it is a model of the slewing capability of the satellite.

Note that this problem is *not* a Travelling Salesman Problem (TSP). The goal of TSP would be to do every window in the minimum amount of time. This goal of this optimization problem is to do the maximum weighted subset of events/windows in a finite duration of time (namely, the start time of the earliest pointing window to the end time of the latest pointing window).

## The Discretized Problem

Instead of considering the continuous range $[st_i, et_i]$ for scheduling an event associated with a particular window, each window is replaced by a discrete set of potential pointing events. A potential pointing event represents the act of pointing at a ground target at an exact time. The times of each event in the discrete set are distributed over $[st_i, et_i]$.

Let $x_{i,j}$ be an indicator variable for the feasibility of changing the satellite's orientation between event $i$ and event $j$ (1 if feasible, 0 if infeasible). Specifically, it is determined by using the agility model to compare whether the required time to reorient, $f_{agility}(\theta_i, \theta_j)$, is less than the available time to reorient, $t_j - (t_i + \tau_i)$.

The discrete optimization problem that yields this highest weighted path can now be written as follows:

$$
\begin{aligned}
&\max_{\bar{c}} \sum_{i=0}^{N} c_i w_i \\
&\textbf{s.t. } (\forall\, i) \ : \\
&\qquad c_i \in 0, 1 \\
&\qquad c_i + c_j \leq 1 \ \ \forall \ x_{i,j} = 0
\end{aligned}
$$

A solution to the discrete problem is in general suboptimal to a solution to the continuous time problem (except in the limit where the number of potential pointing events per window goes to infinity). However, for reasonable numbers of potential pointing events per window the scheduling performance loss is negligable, as the physical agility limitations on the satellite 'saturate' its performance and put a limit on the number of pointing events possible per unit of time.

A benefit of switching to the discrete problem is the ability to solve for the optimal schedule via an exact polynomial time algorithm. The problem can be solved either via dynamic programming or linear programming. The scheduler implemented at Skybox utilizes the former, and a later section describes how the problem is encoded as a graph for solution of the longest path via dynamic programming.

An additional benefit, and a crucial factor for Skybox, is the ability to quickly map the desires of a human user into actions on the edges of the graphs. This allows for a collaborative process utilizing the relative strengths of both automated scheduler and human collection manager.

## Background

The graph-based approach taken at Skybox follows on previous research on modelling spacecraft utilization via graphs(Gabrel and Vanderpooten 2002) (Gabrel 2006) (Bogosian 2008).

An enormous amount of research has taken place in the space of path problems and graph traversal problems. Classic computer science algorithms such as Dijsktra's algorithm (Dijkstra 1959) and the Bellman-Ford algorithm (Bellman 1956) (Ford and Fulkerson 1962) established approaches for calculating the shortest or minimum cost paths through a graph. Research into determining the shortest or minimum cost path under constraints (an NP-hard problem (Garey and Johnson 1979)) followed, and has continued to the present day (Joksch 1966) (Witzgall and Goldman 1965) (Handler and Zang 1980) (Aneja, Aggarwal, and Nair 1983) (Beasley and Christofides 1989) (Hassin 1992) (Mehlhorn and Ziegelmann 2000) (Muhandiramge and Boland 2009) (Storandt 2012).

In contrast to the shortest path problem, the spacecraft scheduling problem described in this paper is a *longest* path problem. The next section describes how the optimization problem is encoded into a graph.

## Directed Acyclic Graphs

As described above, instead of considering each imaging pointing event as occuring somewhere in a continuous range of time $[st_i, et_i]$, each pointing event is replaced by a discrete set of possible pointing events. The times of each possible event in the discrete set are distributed over the range $[st_i, et_i]$.

Now that the total set of potential events is finite, the optimization problem can be modelled as a graph. Each of the graph's nodes represent one of these pointing events for imaging or downlinking data (modelled by a triple, $t_i, \theta_i, \tau_i$). The graph's directed edges encode the ability of the satellite to slew from one event to another, given the amount of time between them, i.e., the presence/absence of an edge between nodes $i$ and $j$ is based on the indicator variable $x_{i,j}$ introduced above. Note that this is a directed *acyclic* graph (DAG) with a natural topological ordering (the chronological order of the nodes based on their values of $t_i$).

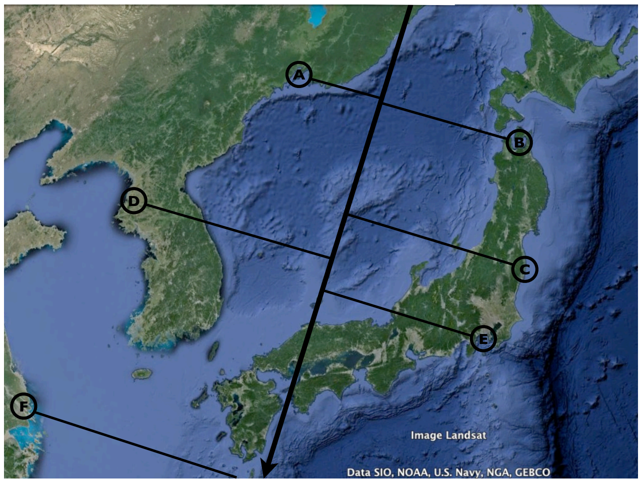

Figure 2: Pointing Opportunities Over Sea of Japan

Figure 2 gives an example of a portion of an orbit for one of Skybox's satellites, showing various ground targets visible over the Sea of Japan. Figure 3 shows the corresponding DAG constructed for this part of the orbit. The nodes are organized in chronological order, with time advancing left to right. In addition to the nodes for the six ground targets, there is also a start node and an end node, which are connected by edges to all of the target nodes in the graph.

For simplicitity, and w.o.l.o.g., only one potential event per window is shown in Figure 3 (i.e., only one node exists per target window). In reality, several potential pointing events/nodes are considered for each window at various event times $t$ in $[st, et]$. This way, the temporal range is still 'explored' and the scheduler is free to consider several actual times during the flyover at which a given ground target could be pointed at for imaging. The amount of pointing events/nodes per target window is a tunable parameter; in the limit as this discretization factor goes to infinity, the optimal solution of the original continuous time problem is recovered.

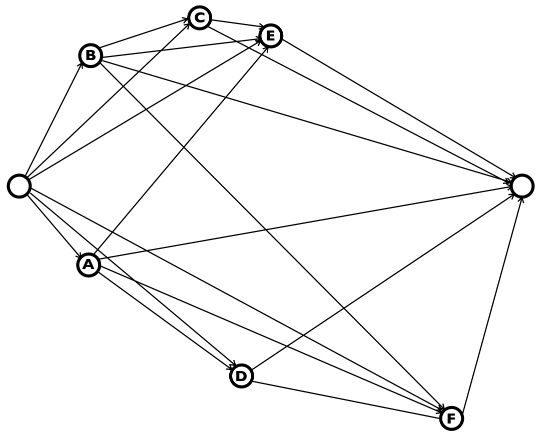

Figure 3: Directed Acyclic Graph For Sea of Japan

## Highest Weighted Path

Importance weights represent the relative priority with which Skybox desires to complete each event. Considering a high priority imaging target $i$ and another, lower priority imaging target $j$, the values of the importance weights will be such that $w_i > w_j$. Importance weights are how business value is quantified and embedded in the optimization problem.

A highest weighted path from the start node to the end node through the graph of Figure 3 represents the optimal set of pointing events the satellite can undertake while flying over the region shown in Figure 2, and is the solution to the discrete optimization problem posed above.

How to calculate lowest/highest weighted path through a DAG has been reviewed exhaustively in the literature (Lawler 1976) (Leiserson et al. 2001); a brief review is given here. Let $W_j$ be the value of the highest weighted path between the start node and node $j$. Let $p_j$ be a pointer to the node that preceeds node $j$ along this highest weighted path. From a node-centric perspective, the optimization at each node $j$ (to choose $W_j$) is:

```
$W_j \leftarrow 0$
for all $\{i \mid x_{i,j} = 1 \, \forall \, j\}$ do
  if $W_i + w_j > W_j$ then
    $W_j \leftarrow W_i + w_j$
    $p_j \leftarrow i$
  end if
end for
```

This optimization is performed for each node in chronological order, from start node to end node. After completing at the end node, backtracking from the end node to the start node via the values of $p_j$ yields the events in the optimal satellite schedule.

In Figure 5, importance weights $w_j$ have been assigned to each event node, and the highest weighted path determined (the edges in bold). Note that the optimal schedule is one that uses high-value events (one of importance weight 5, another of importance weight 3, as compared to the rest of the nodes with importance weight of 1), and *not* one that hits the most events.

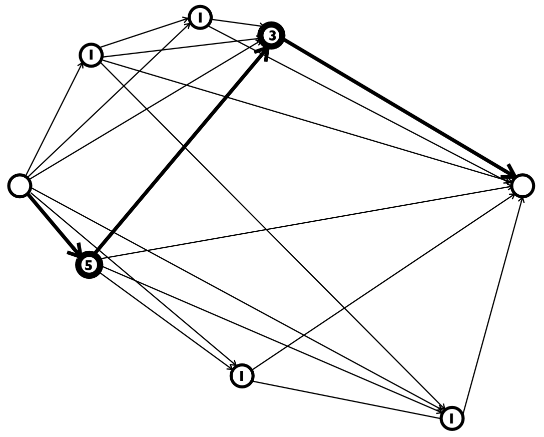

Figure 4: Example of Highest Weighted Path

## Edge Actions

The scheduling system has been designed so that it is possible for a human to interact with this highest weighted path solution and adapt it. The following sections describe how human actions map into changes in the graph and its highest weighted path.

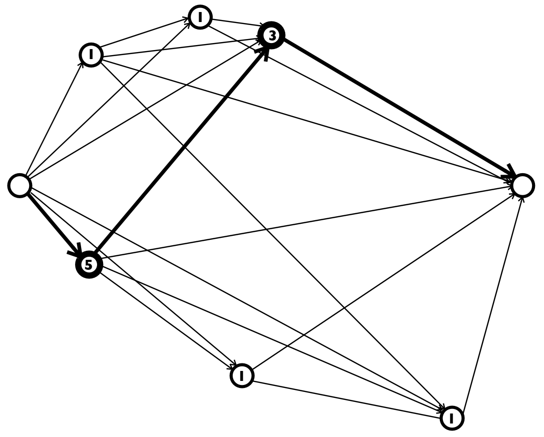

Figure 5: Highest Weighted Path

**Forcing Out** The autonomous scheduler has knowledge of the capabilities of the satellite (via constraint models/edge encodings) as well as relative priority of opportunities with respect to each other (via importance weights). However, when it comes to assessing the presence/absence of certain temporal constraints on imaging, such as cloud cover obscuring a target, the autonomous system may be either unaware or cognitively inferior to a human collection manager. For this reason, the collection manager is given the ability to explicity force a target or targets out of a schedule, so that under no circumstances are any imaging opportunities of that target used at the specified time. This way the satellite does not waste effort taking an image that is going to be cloudy and worthless, and instead replaces that opportunity with other opportunities where the expection of successful imaging is higher.

When a node $j$ is forced out, the action taken is that all edges terminating at node $j$ are disabled (all edges $x_{i,j}$ are set to 0). Figure 6 shows the result of forcing out the node with importance weight 3. Note the removal of all incoming edges, and the new highest weighted path which does *not* pass through this node.

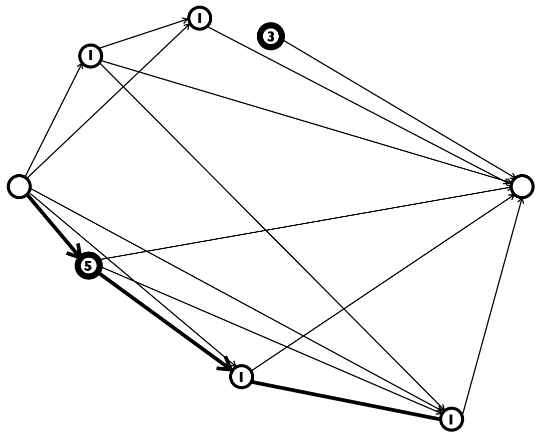

Figure 6: Highest Weighted Path After Node Is Forced Out

**Forcing In** While importance weights $w_i$ are assigned to each target to represent their relative importance, their may be cases where a specific target is of the absolute highest importance and must be included in the schedule of opportunities. An example would be a breaking news event for which HD video from space is desired. Alternatively, an existing low-importance target may temporarily be desired for immediate fulfillment for some reason. As in the force out scenario, a human collection manager is best capable of deciding when a target should be forced in, so that the imaging opportunity is guaranteed to be in the schedule.

When node $j$ is forced in, the action taken is that all edges $x_{i,k}$ "crossing" node $j$ are disabled (all edges $x_{i,k}$ are set to 0, where node $i$ is chronologically earlier than and node $k$ is chronologically later than node $j$). Continuing on from the state of the graph in Figure 6, Figure 7 shows the result of forcing in a node (at the top of the graph) with importance weight 1. Note that all edges running from nodes *earlier than* (left of) the forced in node to nodes *later than* (right of) the forced in node have been removed.

Because of this force in, the highest weighted path no longer passes thru the highest priority target (with importance weight 5). A human forcing in an opportunity is guaranteed to have that opportunity in the resulting schedule, and this trumps importance weight.

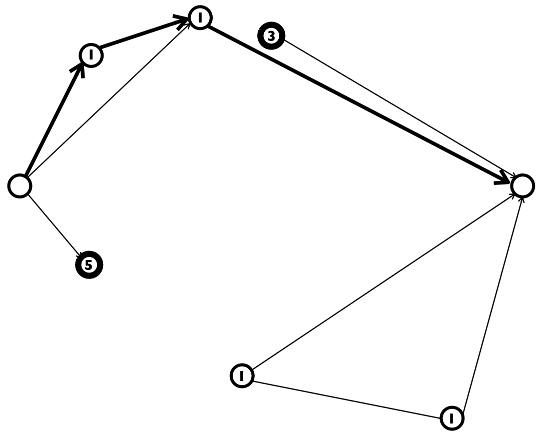

Figure 7: Highest Weighted Path After Node Is Forced In

**Ability Of A Node To Be Forced** There are nodes that will be impossible to force in when other nodes have previously been forced in. This condition is referred to as 'precluded'. It is possible to evaluate a node and determine whether it is precluded, given the current state of the graph. This insight is made visible to the human collection manager, as an aid to collaboration.

Let $precl_i$ be an indicator variable for if node $i$ is precluded or not. Let $N$ be the total number of nodes, and let $i_x$ be the $x$th node in chronological order. The end node is never precluded ($precl_{i_{N-1}}$ is always 1).

The precluded nodes are calculated by doing two passes through the nodes, first in reverse chronological and then in forward chronological order. During the reverse chronological pass, a node is marked as not precluded in any of its successor nodes are not precluded. During the forward chronological pass, a node is marked as precluded if it has no predecessor nodes, or if its all of its predecessor nodes have just been marked as precluded earlier in the forward chronological pass.

```
$precl_{i_{N-1}} \leftarrow 1$

for $\{i = i_{N-2}$ to $i_0\}$ do
    $precl_i \leftarrow 0$
    for all $\{i \mid x_{i,j} = 1 \, \forall \, j\}$ do
        if $precl_j = 1$ then
            $precl_i \leftarrow 1$
        end if
    end for
end for

$M_{NewPrecl} \leftarrow \{\}$
for $\{i = i_1$ to $i_{N-2}\}$ do
    if $precl_i = 1$ then
        $I_{Pred} \leftarrow \{i \mid x_{j,i} = 1 \, \forall \, j\}$
        if $I_{Pred} = \{\}$ or $I_{Pred} \subseteq M_{NewPrecl}$ then
            $precl_i \leftarrow 0$
            $M_{NewPrecl} \leftarrow M_{NewPrecl} \cup i$
        end if
    end if
end for
```

For a node to be not precluded, there must be at least one viable predecessor node and one successor node; the above algorithm determines the nodes that meet these conditions, and those that don't (i.e., the ones that are precluded, $precl_i = 1$).

## Transitive Properties and Space/Time Savings

A property of the DAGs in the satellite scheduling problem is the transitive nature of the edges. As Figure 3 shows, if there is an edge from node $B$ to node $C$ (representing a feasible slew), and there is an edge from node $C$ to node $E$ (representing a feasible slew), then there will be an edge from node $B$ to node $E$, since that slew must also be feasible. That is, the DAG of feasible slews is also the transitive closure of the DAG.

Given a few properties of the satellite scheduling problem, it is observed that such edges (those connecting two nodes for which there is also a connecting path stopping at one or more intermediate nodes) are always dominated and never a part of the longest path through the DAG.

First, the weights $w_i$ are assessed at the nodes, so all edges that terminate at the same node will add the same weight to the overall cumulative weight. In the above example, the act of traversing from node $C$ to node $E$ and the act of traversing from node $B$ to node $E$ provided the same added weight $w_E$.

Second, the weights are non-negative, so the the addition of a preceding edge before node $C$ to node $E$ (e.g., an edge from node $B$ to node $C$) will add weight $w_C \geq 0$. Therefore, the path from node $B$ to node $E$ that stops at intermediate node $C$ will never have less cumulative weight than the direct edge from node $B$ to node $E$ because $w_C + w_E \geq w_E$.

Therefore, as the objective is the highest weighted path through the graph, the skipping or removal of such edges will not affect the optimal result, and doing so saves computational time and storage space. The graph with such dominated edges removed is the transitive reduction of the DAG and can be computed in polynomial time(Aho, Garey, and Ullman 1972).

An even easier time and space reduction is possible using the topology of the nodes. The satellite can physically slew from any orientation to any other orientation in finite time. That is, the agility model $f_{agility}()$ has an upper bound; there is some duration of time $t_{max-slew}$ above which the satellite is guaranteed to be able to slew between two pointing events, no matter the orientation required. Therefore a space reduction is possible by not explicitly enumerating an edge in the graph between two nodes that are separated (chronologically) by more than $t_{max-slew}$. Rather, when optimizing for the highest weighted incoming path to node $j$, the 'for' loop over all incoming edges will consider both explicitly enumerated edges as well as temporarily generated edges from previous nodes that are more than $t_{max-slew}$ in the past. Because of the transitive properties discussed above, a time reduction is possible by not even considering any edges to nodes more than $2t_{max-slew}$ in the past; there will always be a higher weighted, more desirable path into node $j$ than an incoming edge from one of these nodes in the 'far past'.

## Conclusion

An algorithm for scheduling that runs in exact polynomial time is highly desired, because of its speed and bound on generating a deterministic output. Using a graph-based scheduling algorithm like the one described in this paper provides these benefits, as well as allows for human interaction via a set of well-defined actions on edges of the graph. Through these actions, a human operator can force out imaging events that are not desired (e.g., due to cloud cover), force in imaging events that are definitely desired (e.g. breaking news), and be kept aware via a precluded flag on the nodes as to what events are possible or not possible based on other forcing actions the human has taken. Finally, the chronological topology of the directed acyclic graphs used in this algorithm allow for space/time savings in computation.

As Skybox transitions to commercial operations with its imaging satellites, these algorithms are crucial to maximizing the revenue generated from the company's constellation of satellites, and avoiding wasted imaging time.

## Acknowledgements

The author wishes to thank Skybox Imaging and its investors for funding this innovative venture. The author also wishes to thank several people at Skybox for useful discussions about the automated scheduler and its implementation in software : Julian Mann, Shaunak Kashyap, Ollie Guinan, Zeal Jaganatha, Sean Blaes, Matt Debski, and Jon Zemel. Finally, the author thanks Lukas Kuhn at Zenhavior for his formatting and presentation advice.

## References

Aho, A. V.; Garey, M. R.; and Ullman, J. D. 1972. The transitive reduction of a directed graph. *SIAM Journal on Computing* 1(2):131–137.

Aneja, Y. P.; Aggarwal, V.; and Nair, K. P. 1983. Shortest chain subject to side constraints. *Networks* 13(2):295–302.

Beasley, J., and Christofides, N. 1989. An algorithm for the resource constrained shortest path problem. *Networks* 19(4):379–394.

Bellman, R. 1956. On a routing problem. *NOTES* 16(1).

Bogosian, J. 2008. Image collection optimization in the design and operation of lightweight, low areal-density space telescopes. Master's thesis, Massachusetts Institute of Technology.

Dijkstra, E. 1959. A note on two problems in connexion with graphs. *Numerische mathematik* 1(1):269–271.

Ford, L., and Fulkerson, D. 1962. Flows in networks.

Gabrel, V., and Vanderpooten, D. 2002. Enumeration and interactive selection of efficient paths in a multiple criteria graph for scheduling an earth observing satellite. *European Journal of Operational Research* 139(3):533–542.

Gabrel, V. 2006. Strengthened 0-1 linear formulation for the daily satellite mission planning. *Journal of combinatorial optimization* 11(3):341–346.

Garey, M. R., and Johnson, D. S. 1979. Computers and intractability: A guide to the theory of np-completeness.

Handler, G. Y., and Zang, I. 1980. A dual algorithm for the constrained shortest path problem. *Networks* 10(4):293–309.

Hassin, R. 1992. Approximation schemes for the restricted shortest path problem. *Mathematics of Operations Research* 17(1):36–42.

Joksch, H. C. 1966. The shortest route problem with constraints. *Journal of Mathematical analysis and applications* 14(2):191–197.

Lawler, E. L. 1976. Combinatorial optimization: Networks and matroids. 1976. *Holt, Rinehart and Winston, New York* 3.

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

Mehlhorn, K., and Ziegelmann, M. 2000. Resource constrained shortest paths. In *Algorithms-ESA 2000*. Springer. 326–337.

Muhandiramge, R., and Boland, N. 2009. Simultaneous solution of lagrangean dual problems interleaved with preprocessing for the weight constrained shortest path problem. *Networks* 53(4):358–381.

Storandt, S. 2012. Route planning for bicycles-exact constrained shortest paths made practical via contraction hierarchy. In *ICAPS*, volume 4, 46.

Witzgall, C., and Goldman, A. 1965. Most profitable routing before maintenance. In *27th National ORSA Meeting, Boston, MA, USA. Abstract given in ORSA Bulletin*, volume 13.