Large Scale Transportation Service Network Design:
Models, Algorithms and Applications

by

Daeki Kim

B. Arch., Korea University (1987)
M.U.P., University of Kansas (1989)
S.M., Massachusetts Institute of Technology (1994)

Submitted to the Department of Civil and Environmental Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Transportation and Logistics Systems

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1997

©

The author hereby grants to Massachusetts Institute of Technology permission to
reproduce and
to distribute copies of this thesis document in whole or in part.

Signature of Author

Department of Civil and Environmental Engineering
April 29th, 1997

Certified by

Cynthia Barnhart
Associate Professor, Department of Civil and Environmental Engineering
Thesis Supervisor

Accepted by

Joseph M. Sussman
Chairperson, Department Committee on Graduate Students

JUN 24 1997
ARCHIVES
Large Scale Transportation Service Network Design:  
Models, Algorithms and Applications  

by  
Daeki Kim  

Submitted to the Department of Civil and Environmental Engineering  
on April 29th, 1997, in partial fulfillment of the  
requirements for the degree of  
Doctor of Philosophy  
in Transportation and Logistics Systems  

Abstract  
The focus of this research is to solve large scale service network design problems for transportation carriers. Service network design problems arising at railroads, airlines, trucking firms, intermodal partnerships, etc. require the determination of the cost minimizing or profit maximizing set of services and their schedules, given resource constraints. Examples of service network design problems include determining the set of flights and their schedules for an airline; determining the routing and scheduling of tractors and trailers in a trucking operation; and jointly determining the aircraft flights, ground vehicle and package routes and schedules for a provider of express shipment service.  

We have developed a representative, solvable model for large scale transportation service network design problems with time windows. We present three different but equivalent service network design models: a node-arc formulation, a path formulation and a tree formulation. With the use of route based decision variables, we capture complex cost structures and operating regulations and policies. The poor LP bounds limit our ability to solve the problem, so we strengthen our LP relaxation by adding valid inequalities.  

We have developed an algorithm and implementation that finds quality solutions to service network design problems containing hundreds of thousands of constraints and billions of variables. Our optimization approach synthesizes column and row generation techniques and solves the problems with only a small fraction of the constraint matrix.  

We applied our models and solution algorithms to a multimodal express package shipment application. By exploiting special problem structure using a time-line network representation and applying a series of novel problem reduction methods including derived schedules, and node and link consolidation, we achieved dramatic decreases in problem size without loosing exactness of the model.  

Our approach provides quality feasible solutions, results in annual operating cost savings of tens of millions of dollars, reduces the fleet size required, decreases dramatically the time to develop operating plans, and provides planners and analysts with scenario analysis capabilities.  

Thesis Supervisor: Cynthia Barnhart  
Title: Associate Professor, Department of Civil and Environmental Engineering
Acknowledgments

More than four and half years has passed since I joined MIT. For those years, I have been greatly indebted to many persons and I would like to acknowledge and thank them.

My biggest and greatest thank goes to my thesis and academic advisor, Professor Cynthia Barnhart for her constant support, advice, encouragement and friendship. I am always surprised by watching her unlimited inspiration and energy to work. I would like to thank her for introducing me to the world of large scale transportation network optimization. I would like to express my deepest gratitude to my doctoral committee members, Professor Thomas L. Maganti and Professor Yosef Sheffi, for their invaluable guidance, insights and suggestions.

I am greatly indebted to all faculty members at the Center for Transportation Studies for showering me with their excellent education. I would like to thank members of airline operation research group in United Parcel Service (UPS), Mr. Greg Reinhardt, Dr. Keith Ware and a former employee Dr. Jim Yang, for being interested in and supportive of my research. I thank you for useful discussions and prompt technical supports. I would like to thank Professor Pamela H. Vance of Auburn University and Professor Bernard Gendron of University of Montreal, who have been the invaluable source of help.

I would like to thank my friends including Dr. Christopher G. Caplice, Francisco J. Jauffred, Dr. Rajesh G. Shenoi, Dr. Jai Kue Park, Dr. Oh Kyung Kwon, Niranjan Krishnan and members of the Large Scale Optimization Group (LSOG). Their friendship and encouragement made previous years more enjoyable and the tough times more bearable. Unforgettable thank goes to my friends, Chris and Francisco, who helped me in relieving my extreme nerves and worries about completing the dissertation.

This would be incomplete without mentioning the following person who is the best part of my life. I would like to thank my wife, Soojeong Lee, for her endurance, supports and love. I thank you for everything.

I would like to dedicate this dissertation to my Father and Mother. Without their continuous encouragement and supports, I would not have made it through MIT. I love you.
Contents

1 Transportation Service Network Design ................................................. 11
  1.1 Introduction .................................................................................. 11
  1.2 Outline of Thesis ......................................................................... 13

2 Service Network Design Models and Solution Approaches ...................... 14
  2.1 Service Network Design Models .................................................... 14
    2.1.1 Node-Arc Formulation .......................................................... 15
    2.1.2 Path Formulation .................................................................. 17
    2.1.3 Tree Formulation ................................................................ 18
    2.1.4 Model Comparisons .............................................................. 19
  2.2 Valid Inequalities .......................................................................... 20
    2.2.1 Chvátal-Gomory Cuts .......................................................... 20
    2.2.2 Cutset Inequalities ............................................................... 21
  2.3 The Approximate Service Network Design Cut Model ....................... 26
    2.3.1 Motivation ............................................................................ 26
    2.3.2 Approximate SNDP-Cut Formulation .................................... 28
  2.4 Network Representation .................................................................. 31
  2.5 Service Network Design Problem Solutions .................................... 33
    2.5.1 LP Solutions ........................................................................ 34
    2.5.2 IP Solutions ......................................................................... 38
  2.6 Literature Review .......................................................................... 39
    2.6.1 Multimodal Express Package Delivery Problems .................... 39
2.6.2 Network Design Problems ........................................ 40
2.6.3 Multicommodity Flow Problems ................................. 43
2.6.4 Facility Location Problems ...................................... 44
2.6.5 Vehicle Routing Problems ....................................... 45

3 A Multimodal Express Package Shipment Operation 50
  3.1 Problem Description ............................................... 50
    3.1.1 Service Network and Package Flows ....................... 51
    3.1.2 Costs ....................................................... 53
  3.2 The Planning Problem ............................................. 54
    3.2.1 Problem Scope ............................................. 55

4 The Service Network .................................................. 57
  4.1 Network Representation ......................................... 57
  4.2 Network Reduction Methods ..................................... 65

5 Package Flow Models and Solutions ................................. 67
  5.1 Three MCF Formulations: Node-Arc, Path and Tree .......... 67
    5.1.1 Node-Arc Formulation ..................................... 68
    5.1.2 Node-Arc Solution ......................................... 69
    5.1.3 Path Formulation .......................................... 69
    5.1.4 Path Solution ............................................ 70
    5.1.5 Tree Formulation .......................................... 71
    5.1.6 Tree Solution ............................................ 73
  5.2 Computational Results ........................................... 78

6 Service Network Design for Express Package Delivery .......... 80
  6.1 Express Package Delivery: Baseline Model and Solution Approach .......... 80
    6.1.1 Side Constraints ........................................... 81
    6.1.2 Express Package Service Network Design Models .......... 84
    6.1.3 Express Package Service Network Design Model Size .......... 86
    6.1.4 Express Package Service Network Design Solution .......... 88
6.1.5 Express Package Service Network Design Computational Results ........... 90
6.2 Express Package Delivery: Enhanced Model and Solution Approach ......... 90
  6.2.1 Cutset Inequalities ........................................ 90
  6.2.2 EPSND-Cut Model ............................................ 91
  6.2.3 EPSND-Cut Model Size ....................................... 92
  6.2.4 EPSND-Cut Solution .......................................... 93
  6.2.5 EPSND-Cut Computational Results ............................ 94
6.3 Express Package Delivery: Approximate Model and Heuristic Solution Approach 95
  6.3.1 Approximate EPSND-Cut Model ................................ 96
  6.3.2 AEPSND-Cut Solution ......................................... 97
  6.3.3 Bounds on the Heuristic Solution ................................ 97
  6.3.4 Heuristic Computational Experiences .......................... 99
6.4 Scenario Analysis .................................................. 100

7 Contributions and Future Directions ........................................ 107
  7.1 Contributions .................................................... 107
  7.2 Future Directions .................................................. 109

A Service Design Problem Notations ......................................... 111
  A.1 Sets ............................................................. 111
  A.2 Decision Variables ............................................... 112
  A.3 Parameters ....................................................... 112
  A.4 Indicators ......................................................... 113
List of Figures

2-1  Service Network Design Vs. Network Design Solution  15
2-2  Sample Network: Valid Inequalities  24
2-3  Counter Example of LP Equivalence (Undirected Network)  29
2-4  Counter Example of LP Equivalence (Directed Network)  29
2-5  Time-Line Network  32
2-6  Connection Network  32
2-7  Node Consolidation of Time-Line Network  33
2-8  Column Generation Procedure  34
2-9  Cut Generation Procedure  36
2-10 Synchronized Column and Cut Generation  37
2-11 Restricted Master Problem Changes  37

3-1  Express Package Shipment Operation  52
3-2  Research Scope  56

4-1  Derived Pickup Network  58
4-2  Derived Delivery Network  58
4-3  Derived Pickup Package Network with Two Fleet Types  63
4-4  Derived Delivery Package Network with Two Fleet Types  63
4-5  Node Consolidation  65
4-6  Link Consolidation  66

5-1  Random Network Example  76
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6-1</td>
<td>Hub Sorting Capacity Limit Example</td>
<td>83</td>
</tr>
<tr>
<td>6-2</td>
<td>Root Node LP Solution Procedure</td>
<td>89</td>
</tr>
<tr>
<td>6-3</td>
<td>EPSND-Cut LP Solution Algorithm</td>
<td>94</td>
</tr>
<tr>
<td>6-4</td>
<td>Heuristic Solution Approach</td>
<td>95</td>
</tr>
<tr>
<td>6-5</td>
<td>AEPSND-Cut LP Solution Procedure</td>
<td>98</td>
</tr>
</tbody>
</table>
# List of Tables

2.1 Flow Balance Constraints Size Comparison ........................................ 19  
2.2 Uncapacitated Network Design Applications ..................................... 48  
2.3 Capacitated Network Design Applications ......................................... 49  
4.1 Possible Pickup and Delivery Fleet Routes ....................................... 64  
5.1 Deriving a Tree Solution from a Feasible Path Solution ...................... 75  
5.2 Path vs. Tree Formulation ............................................................ 77  
5.3 MCF Data Set Size ............................................................................ 78  
5.4 MCF Matrix Size ............................................................................... 78  
6.1 Data Set Size .................................................................................... 87  
6.2 Number of Constraints in DS3 .......................................................... 87  
6.3 Node and Link Consolidation Results ............................................... 88  
6.4 Cutset Inequality Size ....................................................................... 93  
6.5 Computational Results ....................................................................... 99  
6.6 Scenario Analysis: Two Fleet Types and Three Hubs ......................... 102  
6.7 Scenario Analysis: Single Fleet Type and Single Hub ......................... 102  
6.8 Scenario Analysis: Two Fleet Types and Single Hub ............................ 103  
6.9 Scenario Analysis: Single Fleet Type and Three Hubs ....................... 103  
6.10 Scenario Analysis: Number of Columns, Rows Generated ................. 105  
6.11 Scenario Analysis: Percentage of Demand Through Each Hub, Number of Aircraft Used .............................................................. 106
Chapter 1

Transportation Service Network Design

1.1 Introduction

The focus of our research is to develop models and solution algorithms for service network design problems faced by transportation carriers. Service network design problems arising at railroads, airlines, trucking firms, intermodal partnerships, etc. require the determination of the cost minimizing or profit maximizing set of services and their schedules, given limited resources. Examples of service network design problems include determining the set of flights and their schedules for an airline; determining the routing and scheduling of tractors and trailers in a trucking operation; and jointly determining the aircraft flights, ground vehicle and package routes and schedules for a provider of express shipment service. Our objective is to develop models and solution procedures for these service network design problems in order to generate quality solutions and to address strategic planning issues. We want to use our procedure to answer questions such as:

- Which services should be offered (and at what times) such that available resources are best utilized and profits are maximized?
- What is the best location and size of terminals such that overall costs are minimized?
• What is the best fleet composition such that service requirements are met and profits are maximized?

The model typically contains two types of decision variables, one modeling discrete design variables and the other modeling continuous flow variables, to form a mixed integer program (MIP).

There are two major shortcomings of current service network design models when applied to the problems arising in the transportation industry. First, the interactions between design variables in transportation applications are more complicated than in many other application areas. For example, selecting service between two points implies that a vehicle of some type departs some location and arrives at another. This, in addition, implies that other services must be selected to ensure that flow balance is achieved for that vehicle. Another complexity resulting from these interactions involves fixed costs. Since multiple services may be performed by a single vehicle, fixed costs are associated with sets of design variables and not a single design variable. The second major issue associated with transportation applications is size. In transportation related scheduling problems, time and space are essential ingredients so that the network size explodes when we explicitly consider time and space. State-of-the-art service network design methods simply are not designed for problems of the immense size encountered in transportation. Our focus, then, is to develop a general modeling and solution approach for large scale transportation service network design problems.

We provide novel network reduction methods, which reduce model size without compromising optimality. To further reduce model size and again, maintain optimality, we dynamically generate decision variables and constraints, i.e., we generate them only when they are needed. Our enhanced branch-and-bound solution procedure is specifically designed for large scale integer programs containing millions, or even billions, of decision variables. The underlying idea is to find the optimal solution using row generation as well as column generation methods within the framework of branch-and-bound. Finally, we tailor our approach specifically to provide good feasible solutions for a service network design problem arising in a multimodal express package delivery operation.
1.2 Outline of Thesis

In Chapter 2, we present and compare different models for the service network design problem and, given the notoriously weak linear programming bounds of service network design models, we review two types of valid inequalities that can be used for strengthening. We also describe heuristics and optimization algorithms for large-scale service network design problems, focusing on methods using column generation, row generation and branch-and-bound. We provide a review of relevant literature. We detail our modeling and algorithmic developments for a service network design problem involving multimodal express package delivery in Chapters 3 through 6. In Chapter 3, we describe our express package delivery application and in Chapter 4, we demonstrate how we model the express package network. Since the network is extremely large, we develop and apply a series of novel problem reduction methods that shrink network size without compromising optimality of the model. Models and algorithms for determining the optimal routing of packages, given a fixed service network, is described in Chapter 5. In Chapter 6, we develop, implement and evaluate models and solution algorithms for the multimodal express package delivery problem requiring the simultaneous determination of package flows and selection of services. Finally, in Chapter 7, we provide concluding remarks, detailing the contributions of this thesis, and we describe items for future research.
Chapter 2

Service Network Design Models and Solution Approaches

2.1 Service Network Design Models

The objective of the service network design problem (SNDP) is to find a set of design routes that minimizes the total system costs (i.e., the sum of the fixed design costs and variable flow costs) while satisfying all customer demands without violating the capacity of each service leg. In general, the service network design problem (SNDP) is similar to the capacitated network design problem (NDP) except SNDP has added complexity. Compared to NDP, SNDP additionally requires that transportation assets be assigned to design variables and the assignments are constrained by the requirement to achieve balance of these assets. The balance of service types requires the number of entering service design variables at each location to equal the number of exiting service design variables for each service type. For example, Figure 2-1 shows a network that consists of three nodes and two available service types. The capacity and fixed design cost of service type 1 is (1,10) and is (2, 15) for service type 2. Assume that there is a unit demand from node 1 to 2, 1 to 3, 2 to 1, 2 to 3, and 3 to 1 and that there are no variable flow costs involved. The optimal NDP solution, with value of 70, includes service type 2 links (1,2), (2,3), and (3,1) and service type 1 link (2,3). The optimal SNDP solution, however, includes the additional service type 1 links (1,2) and (3,1) to achieve balance by service type. The resulting SNDP solution value is 90.
Figure 2-1: Service Network Design Vs. Network Design Solution

In this chapter, we present three different but equivalent models for service network design problems. All of the models produce the same optimal solution values; however, they contain different number of constraints and decision variables. The result is a potentially significant difference in computational hardware requirements and solution run times.

To facilitate our discussion of the various SNDP models, we summarize the notations in Appendix A.

2.1.1 Node-Arc Formulation

Let $G = (N, A)$ be a directed graph, where $N$ is the node set and $A$ is the arc set. Let $k$ denote the set of commodities $K$ and $b^k$ denote the quantity of commodity $k$ to be shipped from its origin, denoted $O(k)$, to its destination, $D(k)$, for each $k \in K$. A commodity is defined for each origin-destination pair with nonzero flow. The general service network design model contains two types of variables - one modeling integer design decisions and the other continuous flow decisions. Let $u^f$ denote the capacity of service type $f \in F$. Let $y^f_{ij}$ be an integer variable that indicates the number of times an arc $(i, j) \in A$ of service type $f \in F$ is included in the solution. Let $x^k_{ij}$ denote the fraction of $b^k$ on arc $(i, j)$. Let $h^f_{ij}$ be the fixed cost of including arc $(i, j)$ with service type $f$ in the solution once and $c^k_{ij}$ be the cost per unit flow of $k$ along $(i, j) \in A$. Then, the node-arc representation of the service network design problem is:
\begin{align}
\min & \quad \sum_{f \in F} \sum_{(i,j) \in A} h_{ij} y_{ij}^f + \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^k b^k x_{ij}^k \\
\sum_{k \in K} b^k x_{ij}^k & \leq \sum_{f \in F} w^f y_{ij}^f \quad \text{for all } (i,j) \in A \\
\sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{ji}^k &= \begin{cases} 
1 & \text{if } i = O(k) \\
-1 & \text{if } i = D(k) \\
0 & \text{if } i \neq O(k), D(k) 
\end{cases} \quad \text{for all } i \in N, k \in K \\
\sum_{j \in N} y_{ij}^f - \sum_{j \in N} y_{ji}^f &= 0 \quad \text{for all } i \in N, f \in F \\
x_{ij}^k & \geq 0 \quad \text{for all } k \in K, (i,j) \in A
\end{align}

\begin{align}
y_{ij}^f & \geq 0 \text{ and integer} \quad \text{for all } (i,j) \in A, f \in F
\end{align}

The objective function (2.1) is to find the cost minimizing set of services and routing of flows over these services, subject to constraints (2.2) - (2.6). Constraints (2.2), the \textit{forcing constraints}, limit the amount of flow on an arc to its capacity where an arc capacity is determined by the value of the design variables. Constraints (2.3), \textit{flow conservation constraints}, ensure the service of each commodity, from its origin to its destination, is fully satisfied. Constraints (2.4) are the \textit{design balance constraints} that distinguish service network design problems from conventional network design problems. The balance constraints require the number of each service type entering a node to equal the number leaving, for each node in the network. Finally, constraints (2.5) and (2.6) ensure nonnegativity of commodity flows and integrality and nonnegativity of design variables.
2.1.2 Path Formulation

A design route \( r \) is a set of design variables of some type \( f \) that form a sequence that is balanced everywhere except possibility at the start and end of the sequence. We let \( R^f \) represent the set of all design routes for service type \( f \). \( h^f_r = \sum_{i,j \in A} h_{ij} \alpha^r_{ij} \) denotes the cost of design route \( r \) of type \( f \), \( c^k_p = \sum_{i,j \in A} c^k_{ij} \delta^p_{ij} \) is the cost per unit flow of \( k \) along path \( p \), \( \alpha^r_{ij} \) equals 1 if design variable \((i, j)\) is included in design route \( r \) and equals 0 otherwise, \( \delta^p_{ij} \) equals 1 if path \( p \) includes \((i, j)\) and equals 0 otherwise, \( \beta^r_i \) equals 1 if node \( i \in N \) is the start node of design route \( r \), equals -1 if \( i \) is the end node of \( r \) and equals 0 otherwise.

By substitution in \textit{SNDP\_Node-Arc}, we obtain the following equivalent \textit{SNDP\_Path} formulation (for details see Ahuja, et al.[1]):

\[
\text{(SNDP\_Path)}
\]

\[
\min \sum_{r \in R^f} h^f_r y^f_r + \sum_{k \in K} \sum_{p \in P^k} \left( c^k_p \delta^k \right) x^k_p \quad (2.7)
\]

\[
\sum_{k \in K} \sum_{p \in P^k} \left( \delta^p_{ij} \right) x^k_p \leq \sum_{f \in F} \sum_{r \in R^f} u^f r^f \alpha^r_{ij} \quad \text{for all } (i, j) \in A \quad (2.8)
\]

\[
\sum_{p \in P^k} x^k_p = 1 \quad \text{for all } k \in K \quad (2.9)
\]

\[
\sum_{r \in R^f} \beta^r_i y^f_r = 0 \quad \text{for all } i \in N, f \in F \quad (2.10)
\]

\[
x^k_p \geq 0 \quad \text{for all } k \in K, p \in P^k \quad (2.11)
\]

\[
y^f_r \geq 0 \text{ and integer} \quad \text{for all } f \in F, r \in R^f \quad (2.12)
\]

Constraints (2.8) are the forcing constraints for each design variable \((i, j) \in A\). Constraints (2.9) are the equivalent flow conservation constraints that ensure each commodity has its flow assigned completely from origin to destination. Constraints (2.10) enforce balance at the start and end of the design route (balance at intermediate locations is guaranteed by definition).
Finally, constraints (2.11) and (2.12) ensure nonnegativity of flows and integrality and nonnegativity of design routes.

Compared to SNPD _ Node-Arc, the number of flow conservation constraints in SNPD _ Path is reduced from $|N| \times |K|$ to $|K|$ - a reduction that can have a big impact, especially for large-scale applications containing an extensive network and/or nearly complete demand pattern.

2.1.3 Tree Formulation

If arc costs do not vary by commodity, we can reduce the number of flow conservation constraints in the model using an alternative formulation called the tree formulation, presented in Jones[38]. The idea is to aggregate commodities with the same origin into a single super commodity\(^1\). Each super commodity $s_{O(k)}$ in the set $S$ of super commodities corresponds to the set of commodities $k \in K$ originating at $O(k)$. Since each commodity $k \in s_{O(k)}$ may flow along several paths between its origin and its destination, each super commodity $s_{O(k)}$ can flow along several trees, denoted by the set of trees $Q^{s_{O(k)}}$. We let $\Gamma^{t}_{ij}$ equal 1 if tree $q$ contains arc $(i, j)$ and equal 0 otherwise. Each tree $q \in Q^{s_{O(k)}}$ is rooted at $O(k)$ and contains one $O(k)$ to $D(k)$ path, denoted $p^{b}_{q}$, for only those $k \in s_{O(k)}$. The flow on each path in a tree is a constant $w_{q}$, with $0 \leq w_{q} \leq 1$, of $b$ for every commodity $k$ in the tree. Jones, et al.[38] show that whenever the costs are not commodity specific, the optimal solution values for the path and tree formulations are equal and tree solutions are feasible for the path and node-arc formulations (see Chapter 5 for details).

\[(\text{SNPD } \text{Tree})\]

\[\min \sum_{f \in F} \sum_{r \in R} h_{r}^{f} y_{r}^{f} + \sum_{s_{O(k)} \in S} \sum_{q \in Q^{s_{O(k)}}} c_{q}^{s_{O(k)}} w_{q}^{s_{O(k)}}\]  \hspace{1cm} (2.13)

\[\sum_{s_{O(k)} \in S} \sum_{q \in Q^{s_{O(k)}}} \left( \sum_{k \in s_{O(k)}} \Gamma^{q}_{ij} b^{k} \right) w_{q}^{s_{O(k)}} \leq \sum_{f \in F} \sum_{r \in R} \alpha_{ij}^{f} y_{r}^{f} \text{ for all } (i, j) \in A\]  \hspace{1cm} (2.14)

\[\sum_{q \in Q^{s_{O(k)}}} w_{q}^{s_{O(k)}} = 1 \text{ for all } s_{O(k)} \in S\]  \hspace{1cm} (2.15)

\(^1\)Alternatively, super commodities can be aggregated by destination.
\[ \sum_{r \in R^f} \beta_r^i y_r^f = 0 \quad \text{for all } i \in N, \ f \in F \]  
(2.16)

\[ w_q^{s_{O(k)}} \geq 0 \quad \text{for all } q \in Q^{s_{O(k)}}, \text{ for all } s_{O(k)} \in S \]  
(2.17)

\[ y_r^f \geq 0 \text{ and integer} \quad \text{for all } f \in F, \ r \in R_f \]  
(2.18)

Using the tree formulation, we further reduce the number of flow conservation constraints (2.9) from \(|K|\) in the path formulation to \(|S|\), where \(|S|\) corresponds to the number of super commodities, or equivalently, the number of origins.

### 2.1.4 Model Comparisons

<table>
<thead>
<tr>
<th>Flow Balance Constraints</th>
<th>SNDP Node-Arc</th>
<th>SNDP Path</th>
<th>SNDP Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>N</td>
<td>\times</td>
<td>K</td>
</tr>
</tbody>
</table>

Table 2.1: Flow Balance Constraints Size Comparison

In general, path or tree formulations (SNDP Path and SNDP Tree) have several advantages over link (or equivalently node-arc) formulations. First, we can easily incorporate non-linear and flow-dependent link or route costs in path or tree models but not in link models. Second, certain classes of constraints that are difficult, if not impossible, to write for link models are easily defined in path or tree formulations. For example, it is hard (although not impossible) to control the number of arcs a design route contains with link formulations, but it is easy with path or tree formulations since only design routes satisfying this condition are included in the model. Third, the number of constraints in path and tree models is greatly reduced compared to the number in link models. This is important since LP solution time slows with increases in the number of constraints. The disadvantage, however, is that the decrease in the number of constraints comes at a cost of an exponential increase in the number of decision variables. In Chapter 2, we describe some of the solution methods for handling large numbers of decision variables.
variables. Table 2.1 summarizes the differences in the number of flow balance constraints among the three formulations.

2.2 Valid Inequalities

Given values for the design variable vectors, \( y \), which specify capacities on the arcs, the SNDP has a feasible flow of \( b^k \) units from \( O(k) \) to \( D(k) \) only if the capacity of every \( O(k)/D(k) \) cutset is at least \( b^k \). In general, for any feasible solution to the problem, the aggregate capacity across the cutset must be no less than the demand across the cutset. These aggregate capacity demand inequalities (see Magnanti, Mirchandani and Vachani [51] for details) are expressed as:

\[
\sum_{f \in F} \left( u^f Y^f_{S,T} \right) \geq D_{S,T} \quad \text{for all } O/D \text{ cutsets } \{S,T\},
\]  

(2.19)

where we define cutset \( \{S,T\} \) by a partitioning of the node set \( N \) into two nonempty disjoint sets \( S \subset N \) and \( T = N \setminus S \). An arc \((i,j)\) belongs to cutset \( \{S,T\} \) if nodes \( i \) and \( j \) belong to different sets \( S \) and \( T \). Let \( Y^f_{S,T} \) equal the total number of type \( f \) design variables loaded on the cutset arcs, i.e., \( Y^f_{S,T} \) equals \( \sum_{(i,j) \in \{S,T\}} y^f_{ij} \) for the SNDP_Node-Arc formulation and equals \( \sum_{r \in R} \sum_{(i,j) \in \{S,T\}} y^f_{ij} \alpha^r_{ij} \) for both the SNDP_Path and SNDP_Tree formulations. The aggregate demand, \( D_{S,T} \), denotes the demand of all commodities with \( O(k) \in S \) and \( D(k) \in T \) or \( O(k) \in T \) and \( D(k) \in S \).

Inequalities (2.19) are knapsack inequalities that can be strengthened in several ways. One way is to use a simple integer rounding procedure that produces Chvátal-Gomory (C-G) cuts. Alternatively, we can strengthen the inequalities by generating cutset inequalities, detailed in Magnanti, Mirchandani and Vachani [49], [51] and Magnanti and Mirchandani [50].

2.2.1 Chvátal-Gomory Cuts

We can lift the aggregate capacity demand inequalities (2.19) by applying a simple integer rounding to produce Chvátal-Gomory (C-G) cuts (see Nemhauser[55] for further details).

**Proposition 1** Let \( l \) be a particular service type, where \( l \in F \). Then, the following C-G cuts are valid for each \( O/D \) cutset \( \{S,T\} \).
\[
\sum_{f=1}^{F} \left( \left\lceil \frac{u_f}{u^l} \right\rceil Y^f_{S,T} \right) \geq \left\lceil \frac{D_{S,T}}{u^l} \right\rceil \quad l \in \{1, 2, \ldots, F\}, \text{ for all } O/D \text{ cutsets } \{S, T\} \tag{2.20}
\]

**Proof.** From (2.19) we know that \( \sum_{f=1}^{F} \left( \frac{u_f}{u^l} Y^f_{S,T} \right) \geq \frac{D_{S,T}}{u^l} \). Further, since \( \sum_{f=1}^{F} \left( \frac{u_f}{u^l} Y^f_{S,T} \right) \geq \sum_{f=1}^{F} \left( \frac{u_f}{u^l} Y^f_{S,T} \right) \) and \( \sum_{f=1}^{F} \left( \frac{u_f}{u^l} Y^f_{S,T} \right) \) results in an integer value, inequality (2.20) is valid. \( \blacksquare \)

The difficulty with using \( C-G \) cuts is that there are a huge number of them. For example, in the case of \( N \) origin locations and \( F \) service types, the total number of additional \( C-G \) cuts including the aggregate capacity demand inequalities is \( 2 \left( 2^N - 1 \right) \) \( \times \) \( (F + 1) \).

### 2.2.2 Cutset Inequalities

Aggregate capacity demand inequalities (2.19) may be alternatively strengthened by using cutset inequalities. Cutset inequalities, extensively studied by Magnanti, Mirchandani and Vachani [49], [51] and Magnanti and Mirchandani [50], may provide tighter LP lower bounds than \( C-G \) cuts, and generally work better when there are multiple service types. For ease of exposition, we describe cutset inequalities only for the case containing two service types. Magnanti and Mirchandani[50] provide a more complete description of cutset inequalities.

Assuming that there are only two service types available, (2.19) is:

\[
u^1 Y^1_{S,T} + u^2 Y^2_{S,T} \geq D_{S,T} \tag{2.21}
\]

and the subsequent two types of cutset inequalities are:

\[
Y^2_{S,T} + \frac{u^2}{r(D_{S,T}, u^1)} Y^2_{S,T} \geq \left\lceil \frac{D_{S,T}}{u^1} \right\rceil \tag{2.22}
\]

\[
\frac{u^1}{r(D_{S,T}, u^2)} Y^1_{S,T} + Y^2_{S,T} \geq \left\lceil \frac{D_{S,T}}{u^2} \right\rceil \tag{2.23}
\]

where \( r \left(D_{S,T}, u\right) \equiv D_{S,T} - \left(\left\lceil \frac{D_{S,T}}{u} \right\rceil - 1\right) u \).
Proposition 2 Cutset inequalities (2.22) and (2.23) are valid (see also Magnanti, Mirchandani and Vachani[51]).

Proof. Rewriting (2.21) as:

\[ u^2Y_{S,T}^2 \geq D_{S,T} - u^1Y_{S,T}^1 \]

Since \( D_{S,T} = r(D_{S,T}, u^1) + \left( \left\lfloor \frac{D_{S,T}}{u^1} \right\rfloor - 1 \right) u^1 \) and \( u^1 \geq r(D_{S,T}, u^1) \), it follows that

\[
\begin{align*}
    u^2Y_{S,T}^2 &\geq r(D_{S,T}, u^1) + \left( \left\lfloor \frac{D_{S,T}}{u^1} \right\rfloor - 1 \right) u^1 - u^1Y_{S,T}^1 \\
    &= u^1 \left( \left\lfloor \frac{D_{S,T}}{u^1} \right\rfloor - Y_{S,T}^1 - 1 \right) + r(D_{S,T}, u^1) \\
    &\geq r(D_{S,T}, u^1) \left( \left\lfloor \frac{D_{S,T}}{u^1} \right\rfloor - Y_{S,T}^1 - 1 \right) + r(D_{S,T}, u^1) \\
    &= r(D_{S,T}, u^1) \left( \left\lfloor \frac{D_{S,T}}{u^1} \right\rfloor - Y_{S,T}^1 \right)
\end{align*}
\]

Therefore, \( r(D_{S,T}, u^1)Y_{S,T}^1 + u^2Y_{S,T}^2 \geq r(D_{S,T}, u^1) \left\lfloor \frac{D_{S,T}}{u^1} \right\rfloor \) and (2.22) is valid.

Similarly, rewriting (2.21) as:

\[ u^1Y_{S,T}^1 \geq D_{S,T} - u^2Y_{S,T}^2 \]

Since \( D_{S,T} = r(D_{S,T}, u^2) + \left( \left\lfloor \frac{D_{S,T}}{u^2} \right\rfloor - 1 \right) u^2 \) and \( u^2 \geq r(D_{S,T}, u^2) \), it follows that

\[
\begin{align*}
    u^1Y_{S,T}^1 &\geq r(D_{S,T}, u^2) + \left( \left\lfloor \frac{D_{S,T}}{u^2} \right\rfloor - 1 \right) u^2 - u^2Y_{S,T}^2 \\
    &= u^2 \left( \left\lfloor \frac{D_{S,T}}{u^2} \right\rfloor - Y_{S,T}^2 - 1 \right) + r(D_{S,T}, u^2) \\
    &\geq r(D_{S,T}, u^2) \left( \left\lfloor \frac{D_{S,T}}{u^2} \right\rfloor - Y_{S,T}^2 - 1 \right) + r(D_{S,T}, u^2) \\
    &= r(D_{S,T}, u^2) \left( \left\lfloor \frac{D_{S,T}}{u^2} \right\rfloor - Y_{S,T}^2 \right)
\end{align*}
\]

Therefore, \( u^1Y_{S,T}^1 + r(D_{S,T}, u^2)Y_{S,T}^2 \geq r(D_{S,T}, u^2) \left\lfloor \frac{D_{S,T}}{u^2} \right\rfloor \) and (2.23) is valid. □

Proposition 3 Cutset inequalities (2.22) and (2.23) do not dominate C-G cuts (2.20), and vice versa.

Proof. Given two fleet types, C-G cuts (2.20) are:
\[ Y_{S,T}^{1} + \left[ \frac{u^2}{u^1} \right] Y_{S,T}^{2} \geq \left[ \frac{D_{S,T}}{u^1} \right] \]  

(2.24)

\[ Y_{S,T}^{1} + Y_{S,T}^{2} \geq \left[ \frac{D_{S,T}}{u^2} \right] \]  

(2.25)

Other than the following two cases, dominance between cutset inequalities and C-G cuts can not be determined a priori.

**Case 1:** When \( \frac{u^2}{r(D_{S,T}, u^1)} \leq \left[ \frac{u^2}{u^1} \right] \) and \( u^1 < r(D_{S,T}, u^2) \leq u^2 \):

If \( \frac{u^2}{r(D_{S,T}, u^1)} \leq \left[ \frac{u^2}{u^1} \right] \), (2.22) is stronger than (2.24). If \( u^1 < r(D_{S,T}, u^2) \leq u^2 \), \( \frac{u^1}{r(D_{S,T}, u^2)} < 1 \) and (2.23) is stronger than (2.25). Therefore, cutset inequalities dominate C-G cuts.

**Case 2:** When \( \left[ \frac{u^2}{u^1} \right] \leq \frac{u^2}{r(D_{S,T}, u^1)} \) and \( r(D_{S,T}, u^2) \leq u^1 \):

If \( \left[ \frac{u^2}{u^1} \right] \leq \frac{u^2}{r(D_{S,T}, u^1)} \), (2.24) is stronger than or equal to (2.22). If \( r(D_{S,T}, u^2) \leq u^1 \), \( \frac{u^1}{r(D_{S,T}, u^2)} \geq 1 \) and (2.25) is stronger than or equal to (2.23). Therefore, C-G cuts dominate cutset inequalities.

Since in some cases, the C-G cuts dominate the cutset inequalities and in other cases the result is reversed, we use the following mix of C-G cuts and cutset inequalities when the number of service types is two.

\[
\alpha Y_{S,T}^{1} + Y_{S,T}^{2} \geq \left[ \frac{D_{S,T}}{u^1} \right], \quad \alpha \equiv \min \left\{ \left[ \frac{u^1}{u^2} \right], \frac{u^1}{r(D_{S,T}, u^2)} \right\} \\
Y_{S,T}^{1} + \beta Y_{S,T}^{2} \geq \left[ \frac{D_{S,T}}{u^2} \right], \quad \beta \equiv \min \left\{ \left[ \frac{u^2}{u^1} \right], \frac{u^2}{r(D_{S,T}, u^1)} \right\}
\]  

(2.26)

Again, the challenge in using cutset inequalities is the large number of potential inequalities. For example, in the case of \( N \) origin locations and \( F \) service types, the total number of additional cutset inequalities including the aggregate capacity demand inequalities is \( \{2(2^N - 1)\} \ast (F + 1) \), which is the same as the number of C-G cuts. Another difficulty with cutset inequalities is that it becomes computationally expensive to lift them when the number of service types exceeds two. The generalized version of cutset inequalities is found in Magnanti, Mirchandani and Vachani [51].

23
Figure 2-2: Sample Network: Valid Inequalities

We illustrate the C-G cuts and cutset inequalities by working through the following simple example.

**Example 4** Let's assume (see Figure 2-2) that there are only two nodes (denoted 1 and 2) with the demand of 30 from node 1 to 2 and 40 from 2 to 1, respectively. There are two service types available, namely type 1 and type 2, with capacity of 50 and 100, and fixed cost of 10 and 15. The cost of sending one unit of flow of any commodity on arc (1,2) and (2,1) is one.

The corresponding SNDP _Path_ model is:

\[
\begin{align*}
\text{min} & \quad 10y_1^1 + 10y_2^1 + 15y_1^2 + 15y_2^2 + 30x_1^1 + 40x_2^2 \\
\text{subject to} & \quad : \\
-30x_1^1 + 50y_1^1 + 100y_2^2 & \geq 0 \quad (2.27) \\
-40x_1^2 + 50y_2^1 + 100y_2^2 & \geq 0 \quad (2.28)
\end{align*}
\]

\[^2\text{In fact, SDP\_Node-Arc and SDP\_Tree models are identical to SDP\_Path for this problem.}\]
\[ x^1_1 = 1 \] \hspace{1cm} (2.29)

\[ x^2_1 = 1 \] \hspace{1cm} (2.30)

\[ -y^1_1 + y^2_1 = 0 \] \hspace{1cm} (2.31)

\[ -y^2_1 + y^2_2 = 0 \] \hspace{1cm} (2.32)

\[ x^1_1, x^2_1 \geq 0 \] \hspace{1cm} (2.33)

\[ y^1_1, y^2_1, y^2_1, y^2_2 \geq 0 \text{ and integer} \] \hspace{1cm} (2.34)

The optimal IP solution of this problem is 90.0 with \[ x^1_1 = x^2_1 = 1.0 \] and \[ y^1_1 = y^2_1 = 1.0 \] and the LP relaxation results in the objective value of 82.0 with \[ x^1_1 = x^2_1 = 1.0 \] and \[ y^2_1 = y^2_2 = 0.4. \] There exist the following two aggregate capacity demand inequalities:

\[ 50y^1_1 + 100y^2_1 \geq 30 \] \hspace{1cm} (2.35)

\[ 50y^1_2 + 100y^2_2 \geq 40, \] \hspace{1cm} (2.36)

and the following C-G cuts:

\[ y^1_1 + 2y^2_1 \geq 1 \text{ and } y^1_1 + y^2_1 \geq 1 \] \hspace{1cm} (2.37)

\[ y^1_2 + 2y^2_2 \geq 1 \text{ and } y^1_2 + y^2_2 \geq 1. \] \hspace{1cm} (2.38)

When we add the C-G cuts of (2.37) and (2.38) to the LP relaxation of SNDP Path, the
optimal LP solution is the optimal IP solution.

Alternatively, we consider generating cutset inequalities by first constructing $r(D_{1,2}, u^1) = r(30, 50) = 30$, $r(D_{1,2}, u^2) = r(30, 100) = 30$, $r(D_{2,1}, u^1) = r(40, 50) = 40$, and $r(D_{2,1}, u^2) = r(40, 100) = 40$. Cutset inequalities in the form of (2.22) and (2.23) are then:

\[
y_1^1 + \frac{100}{30} y_1^2 \geq 1 \text{ and } \frac{50}{30} y_1^1 + y_1^2 \geq 1 \quad (2.39)
\]

\[
y_2^1 + \frac{100}{40} y_2^2 \geq 1 \text{ and } \frac{50}{40} y_2^1 + y_2^2 \geq 1. \quad (2.40)
\]

When we add the cutset inequalities of (2.39) and (2.40) to the LP relaxation of SNDP-Path, the optimal LP solution is increased from 82.0 to 87.6 with $y_1^1 = y_2^1 = 0.706$, $y_1^2 = y_2^2 = 0.118$ and $x_1^1 = x_2^1 = 1.0$.

The example shows the effectiveness of both C-G cuts and cutset inequalities and favors C-G cuts over cutset inequalities. The strength and effectiveness of either class of valid inequality depends, however, on the particular problem instance.

### 2.3 The Approximate Service Network Design Cut Model

#### 2.3.1 Motivation

When flow variable costs are negligible compared to fixed design variable costs, we can approximate the service network design problem without considering flows explicitly. Magnanti and Mirchandani[50] consider a single O/D three facility loading problem (IP3) with sufficiently large $L$ as:

\[ [\text{P(IP3)}] \]

\[
\min \sum_{(i,j) \in A} (a_{ij} x_{ij} + b_{ij} y_{ij} + c_{ij} z_{ij})
\]
\[ \sum_{i \in N} f_{ji} - \sum_{i \in N} f_{ij} = \begin{cases} -d & \text{if } i = O \\ d & \text{if } i = D \\ 0 & \text{otherwise} \end{cases} \text{ for all } i \in N \]

\[ f_{ij} + f_{ji} \leq x_{ij} + Cy_{ij} + \lambda Cz_{ij} \text{ for all } (i, j) \in A \]

\[
\begin{align*}
x_{ij} & \leq L \\
y_{ij} & \leq L \\
z_{ij} & \leq L
\end{align*}
\text{ for all } (i, j) \in A
\]

\[ x_{ij}, y_{ij}, z_{ij} \in Z^+, \ f_{ij}, f_{ji} \geq 0 \text{ for all } (i, j) \in A \]

\(x, y, z\) denote three different types of facilities with capacity of 1, \(C\) and \(\lambda C\) and unit installation costs of \(a, b\) and \(c\), respectively. There is a single commodity demand of \(d\) from \(O\) to \(D\). \(f_{ij}\) denotes the flow of the commodity from \(i\) to \(j\) on arc \((i, j) \in A\).

Projecting \(IP3\) into the subspace of design variables, the resulting equivalent cutset formulation \(CUT3\) is:

\[ [P(CUT3)] \]

\[
\min \sum_{(i,j) \in A} (a_{ij}x_{ij} + b_{ij}y_{ij} + c_{ij}z_{ij})
\]

\[ X_{S,T} + CY_{S,T} + \lambda CZ_{S,T} \geq D_{S,T} \text{ for all } O/D \text{ cutset } \{S,T\} \]
\[
\begin{align*}
&x_{ij} \leq L \\
&y_{ij} \leq L \\
&z_{ij} \leq L
\end{align*}
\] for all \((i, j) \in A\)

\[x_{ij}, y_{ij}, z_{ij} \in Z^+ \text{ for all } (i, j) \in A,\]

where \(X_{S,T}\) equals the total number of facility type \(x\) loaded on the cutset arcs, i.e., \(X_{S,T} = \sum_{(i,j) \in (S,T)} x_{ij}\). \(Y_{S,T}\) and \(Z_{S,T}\) are similarly defined. \(D_{S,T}\) denotes the total demand from the set \(S\) to the set \(T\) and equals \(d\).

**Proposition 5** The polyhedron defined by the linear programming relaxation of \(P(IP3)\) is the projection of the polyhedron defined by the linear programming relaxation of the \(P(CUT3)\) (Magnanti and Mirchandani[50] and Mirchandani[54])

**Proof.** Given \(\bar{x}, \bar{y}\) and \(\bar{z}\), a feasible solution to \(P(IP3)\), which specify capacities on the arcs, the problem has a feasible flow of \(d\) units from \(O\) to \(D\) if and only if the capacity of every \(O/D\) cutset is at least \(d\). That is, the problem has a feasible flow if and only if the design variables satisfy the aggregate capacity demand inequality, \(\bar{X}_{S,T} + C\bar{Y}_{S,T} + \lambda C\bar{Z}_{S,T} \geq d\) for all \(O/D\) cutsets \(\{S,T\}\). \(\blacksquare\)

### 2.3.2 Approximate SNDP-Cut Formulation

For general multicommodity problems, aggregate capacity demand inequality constraints are a necessary, but not a sufficient, condition to guarantee a feasible multicommodity flow solution (Mirchandani[54]). That is, given decision variable vectors of \(\bar{x}, \bar{y}\) and \(\bar{z}\) that satisfy \(\bar{X}_{S,T} + C\bar{Y}_{S,T} + \lambda C\bar{Z}_{S,T} \geq D_{S,T}\), \(\forall\) \(O/D\) cutsets \(\{S,T\}\), the existence of flow vectors \(\bar{f}^k_{ij}\) for all \(k \in K\) that satisfy flow conservation and forcing constraints is not guaranteed. Mirchandani[54] shows and analyzes very special cases when this LP equivalence holds for the multicommodity case. We provide the following two examples (i.e., one for the undirected network case and the other for the directed network case) where LP equivalence between the two models does not hold.
Example 6 Let’s assume that there are six nodes on an undirected, complete network and there is demand of 2 from node 1 to 6, 1 from 2 to 3 and 1 from 4 to 5, respectively. There is a single service type available, with capacity of one. Figure 2-3 shows one feasible solution that satisfies the aggregate capacity demand constraints of $\bar{X}_{S,T} \geq D_{S,T}$, $\forall \ O/D$ cutsets $\{S,T\}$. Arcs in the graph represent the decision variables whose solution value equal 1. However, the solution does not satisfy the flow conservation constraints for the demands between nodes 2 and 3 and nodes 4 and 5.

\(^3\text{The example is drawn from Mirchandani[54]}\)
Example 7 Let's assume that there are six nodes on a directed network and there is demand of 1 from node 1 to 3 and 1 from 2 to 4, respectively. There is a single service type available, with capacity of one. Figure 2-4 shows one feasible solution that satisfies the aggregate capacity demand constraints. Arcs in the graph represent the decision variables whose solution value equal 1. However, the solution does not satisfy the flow conservation constraints for the demands.

Because SNDP's involve multiple commodities, the LP equivalence between the flow and cut models does not hold. However, we can approximate SNDP-Node-Arc with ASNDP-Cut_Node-Arc and SNDP_Path and SNDP_Tree with ASNDP-Cut_Route. The ASNDP-Cut_Route approximation model allows us to solve SNDP approximately considering only design set variables and ignoring the huge number of flow variables and the large number of constraints associated with the flow variables. \( y_{ij}^f \) equals \( \sum_{(i,j) \in \{S,T\}} y_{ij}^f \) for ASNDP-Cut_Node-Arc and equals \( \sum_{r \in R^f} \sum_{(i,j) \in \{S,T\}} y_{ij}^f \alpha_{ij}^r \) for ASNDP-Cut_Route.

\[(\text{ASNDP-Cut_Node-Arc})\]

\[
\min \sum_{f \in F} \sum_{(i,j) \in A} h_{ij}^f y_{ij}^f
\]

\[
\sum_{f \in F} u^f Y_{S,T}^f \geq D_{S,T} \quad \text{for all } O/D \text{ cutset } \{S,T\}
\]

\[
\sum_{j \in N} y_{ij}^f - \sum_{j \in N} y_{ji}^f = 0 \quad \text{for all } i \in N, f \in F
\]

\[
y_{ij}^f \geq 0 \text{ and integer} \quad \text{for all } (i,j) \in A, f \in F
\]

\[(\text{ASNDP-Cut_Route})\]
\[
\min \sum_{f \in F} \sum_{r \in R'} h_f^r y_f^r
\]

\[
\sum_{f \in F} u_f^r Y_{S,T}^f \geq D_{S,T} \quad \text{for all } O/D \text{ cutset } \{S, T\}
\]

\[
\sum_{r \in R'} \beta_i^r y_f^r = 0 \quad \text{for all } i \in N, , f \in F
\]

\[y_f^r \geq 0 \text{ and integer} \quad \text{for all } f \in F, r \in R'.\]

### 2.4 Network Representation

In many transportation-related scheduling problems, time and space are essential elements. In order to evaluate properly the system, the conventional modeling practice is to build dynamic models. At their core, these models have a time-space structure, with nodes representing time and space and arcs representing movement in time and possibly, space. Two different representations of time-space networks for service design are the connection network and the time-line network.

In both the time-line and connection networks, a node \(i\) represents a location, denoted \(l_i\), and a time, denoted \(t_i\), and there are two types of arcs. In the time-line and the connection network, a "design" arc is placed between every pair of nodes \(i\) and \(j\) if a design variable from location \(l_i\) to location \(l_j\) with elapsed time \(t_j - t_i\) is possible. The second type of arc in the time-line network, the "time-line" arc, exists between every pair of adjacent nodes at the same locations, where node \(j\) is adjacent to node \(i\) if \(t_j - t_i \geq 0, j \neq i\), and \(t_{j'} - t_i \geq t_j - t_i, \ \forall \ j' \neq i\). The arcs represent the movement in time at a location (see Figure 2-5). The second type of arc in the connection network, the "connection" arcs, potentially exists between every pair of...
node: i and j at the same location (see Figure 2-6).

The advantages and disadvantages of the two network types can be summarized as follows:

- The time-line network usually requires fewer arcs than the connection network; and

- The connection network is better able to model complicating constraints, such as restrictions on sets of design variables selected.

The best network to use, then, is problem specific.

We can reduce dramatically the size of time-line networks using a network reduction method, called node consolidation. Without destroying feasibility of any path in the network, we consider all nodes representing some location i and arrange the nodes in increasing order of their associated time. We aggregate each sequence of adjacent arrival nodes (departure nodes) that are uninterrupted by a departure node (an arrival node), into a single super arrival (departure)
node. Then, we merge a super arrival node followed by a super departure node into a single node, called a super node. Figure 2-7 shows how the network of Figure 2-5 is represented after node consolidation. The node consolidation method has been previously applied by Hane, et al.[34], Nemhauser[57] and Shenoi[67] in applications for the airline industry. Another network reduction method, referred to as link consolidation, is detailed in Chapter 4.

2.5 Service Network Design Problem Solutions

The typical size of real-world service network design problems is prohibitively huge so that direct solution is usually unachievable even with state-of-the-art LP/IP solvers (such as, CPLEX[20], OSL[36], MINTO[56], etc.). The number of constraints in the path and tree SNDP models is greatly reduced compared to the number in the link model, however, the number of decision variables is increased exponentially. In order to achieve tight LP relaxation, we add valid inequalities and result in exponential number of constraints (see SNDP-Cut).

In this section, we introduce solution methods that are capable of solving huge integer programming problems (IP's). We use branch-and-bound (see, Bradley, Hax and Magnanti[8] and Nemhauser and Wolsey[55]) in order to obtain integer solutions by solving a series of LP subproblems. Since each LP may be very large, we introduce solution methods that are capable of solving huge linear programming problems. Our aim is to overview solution methods, without providing details of the algorithms. More detailed descriptions are provided in Bradley, Hax and Magnanti[8], Nemhauser and Wolsey[55], Barnhart, et al.[11] and Desrosiers, et al.[26].
2.5.1 LP Solutions

Column Generation

Column generation methods achieve optimal solutions to LP's containing a huge number of decision variables, without explicitly considering all decision variables. We start with a very small number of decision variables, called the restricted master problem (RMP), and eliminate the remaining variables in the original formulation, called master problem (MP). To check the optimality with regard to the MP of a solution to the RMP, a subproblem, called the pricing problem, is solved. The outcome is that either optimality is proved or previously ignored columns are identified. If such columns are found, we add the new columns to the RMP and the RMP is re-optimized. The entire process is repeated until the subproblem finds no more columns to add and optimality of the MP is proved. Figure 2-8 shows the general steps of column generation. In the worst case, all variables may be added and the huge MP is solved at the last step, however.

To start column generation, we may initially introduce artificial variables with arbitrarily
large costs to ensure feasibility of the RMP at each iteration. If the MP is feasible, artificial variables will not be in an optimal solution. If some artificial variables are in the optimal solution, the MP is infeasible.

The difficulty of solving the pricing subproblem determines the tractability of applying column generation approaches, since the subproblem is solved hundreds or thousands of times. *Implicit column generation* refers to solving the pricing subproblem without evaluating all columns (e.g., by solving a shortest path problem), while *explicit column generation* refers to solving the pricing subproblem by computing the reduced cost of possibly all variables.

**Cut Generation**

Cut generation methods achieve optimal solutions to LP’s with huge numbers of constraints or achieve better LP relaxations (i.e., an improved bound on the optimal objective function value) by selectively adding valid inequalities to the LP relaxation. The methods start with a small number of constraints, called the relaxed problem (*RP*), eliminating a large set of constraints contained in the original formulation (the master problem *MP*). By ignoring some constraints and/or valid inequalities, the RP optimal solution may be LP infeasible or may be a weak bound on the optimal IP solution. To check the optimality of the current solution, a subproblem, called the *separation problem*, is solved to identify violated constraints or valid inequalities (i.e., cuts) to add to the current RP. If one or more cuts are found, they are added to the RP and the RP is re-optimized. The entire process is repeated until the separation problem finds no violated constraints and/or a sufficiently tight bound is achieved. Figure 2-9 depicts the steps of cut generation algorithms.

*Explicit cut generation* is necessary when the separation problem cannot be solved efficiently. In this case, violated inequalities may be identified by evaluating the possibility of each constraint, given RP optimal solution. *Implicit cut generation* refers to the case when the separation problem can identify violated constraints without evaluating each one individually.

**Synchronized Column and Cut Generation**

When a LP contains both huge number of constraints and a huge number of variables, it might be necessary to apply column and cut generation. These methods start with a small number
of columns and constraints from the original master problem MP, to form the restricted master problem \( RMP \). Column generation might be applied first\(^4\) and the process is repeated until all columns have non-negative reduced costs or some stopping criterion is satisfied. Then, the separation problem is repeatedly solved until no violated inequalities are identified or until some stopping criterion is satisfied. After completing cut generation, we repeat the entire procedure. The algorithm terminates with a LP optimal solution when there are no columns or violated constraints to add.

Figure 2-10 shows the steps of synchronized column and cut generation algorithm and Figure 2-11 shows how our \( RMP \) changes as the algorithm of Figure 2-10 progresses. Implementing synchronized column and cut generation procedures is a nontrivial task. One challenge is in adding constraints that do not change the solution procedure for the pricing subproblems and another is adding variables that do not change the solution procedure for generating violated inequalities. The pricing (separation) problem can be much harder after additional cuts (columns)

\(^4\)Alternatively, cut generation can be performed first.
Figure 2-10: Synchronized Column and Cut Generation

Figure 2-11: Restricted Master Problem Changes
are added, because the new rows (columns) can destroy the structure of the pricing (separation) problem. This compatibility issue may be avoided when we apply explicit column generation and explicit cut generation provided that explicit column and cut generation procedures are computationally inexpensive.

2.5.2 IP Solutions

*Branch-and-bound* (B&B) is essentially a strategy of *divide and conquer* in order to obtain integer solutions by solving a series of LP subproblems (Bradley, Hax and Magnanti[8]). When the problem contains a huge number of decision variables, a modified form of B&B, called *branch-and-price* (B&P), is involved. B&P uses column generation in solving each LP at each node of the B&B tree. Branching occurs when no columns price out to enter the basis and the LP solution does not satisfy the integrality conditions. The major challenge of B&P is to maintain *compatibility* between the rules used for branching and the pricing problem (Barnhart, et al.[11]). This means that a branching rule should be enforceable without changing the structure of the pricing problem since LP’s at each of the new branches are solved using column generation. For example, when the binary decision variables \( x \) are path-based, enforcing \( x_j = 0 \) requires a special type of network representation in order not to destroy tractability of the subproblem. Moreover, branching strategy based on variable dichotomy is *unbalanced* since setting \( x_j = 1 \) forces many other \( x_k = 0 \), for \( k \neq j \); whereas setting \( x_j = 0 \) has much less impact since there are a huge number of decision variables. Parker and Ryan[61] suggest an alternative one efficient branching strategy, as do Desrochers and Soumis[25], Desrosiers, Solomon and Soumis[26], Barnhart, et al.[11] and Vance, et al.[72]

When the problem contains a huge number of constraints (or equivalently valid inequalities), another variant of B&B, called *branch-and-cut* (B&C), can be used. B&C uses uses cut generation in solving each LP at each node of the B&B tree. Branching occurs when no violated inequalities are found or some stopping criteria is satisfied. The difficulties of B&C are similar to those of B&P.

When the problem contains both a huge number of columns and a huge number of valid inequalities, branch-and-price and branch-and-cut can be combined to form a *branch-and-price-and-cut* (B&P&C) solution procedure. There are fundamental challenges in obtaining optimal
(or even reasonably good feasible) solutions of huge IP's with B&P&C. In addition to the
difficulties arising from B&P and B&C individually, the principal difficulty of B&P&C is incompatibility between column generation and cut generation. The pricing (separation) problem can be much harder after additional cuts (columns) are added, because the new rows (columns) can destroy the structure of the pricing (separation) problem. Some heuristic solution approaches (Barnhart, et al.[11]) solve the root node LP relaxation by synchronized column and cut generation and solve the remaining LP's with B&B using only the columns and constraints generated at the root node. When there are a sufficient number of columns and valid inequalities generated at the root node, the heuristic may be able to generate good feasible solution (Barnhart, et al.[11]). However, good or even feasible solutions to the problem is not guaranteed.

2.6 Literature Review

A variety of applications in transportation, distribution, communication, and several other problem domains require trade-offs between variable operating costs and fixed design costs for providing service. Although the literature is extensive, limited capability exists to solve large-scale transportation applications. Interestingly, an impressive range of transportation related problems, namely the network design problem, the multicommodity flow problem, the facility location problem, and the vehicle routing problem, can be thought of as variations or special cases of the SNDP.

2.6.1 Multimodal Express Package Delivery Problems

Considering multimodal express package delivery problems specifically, Barnhart and Schneur[12] developed a model and algorithm to solve an express shipment service network design problem. In their model, however, (1) there is only one hub, (2) transfer of shipments between aircraft at gateways is not allowed, (3) only one type of aircraft is allowed to serve a gateway, and (4) there is only one feeder service, namely ground vehicle. The result is that shipment routings are determined completely by aircraft routes. Kuby and Gray[40] examine the effectiveness of hub-and-spoke networks with stopovers and feeders and compare their performance to direct flights to the hub, with an application to Federal Express. There are package flow decision
variables as well as aircraft routing decision variables. They assume that one sorting hub exists and the network size is relatively small: they consider only the western United States in order to avoid computational difficulties. Very recently, Pandit[60] solves an air-cargo distribution system problem as a multi-mode, multi-hub (one air hub and one ground hub) network design problem, without scheduling requirements and hub sort capacity restrictions.

2.6.2 Network Design Problems

The $SNDP_{Node-Arc}$ formulation without design balance constraints and with binary design variables is a multilevel network design problem (see Balakrishnan, Magnanti and Mirchandani[5]). When there is only a single service type, it becomes the network design problem ($NDP$).

\begin{equation}
\begin{aligned}
\min & \sum_{(i,j)\in A} h_{ij}y_{ij} + \sum_{k\in K} \sum_{(i,j)\in A} c_{ij}^k x_{ij}^k \\
\text{s.t.} & \sum_{k\in K} x_{ij}^k \leq u_{ij}y_{ij} \quad \text{for all } (i,j) \in A \\
\end{aligned}
\end{equation}

\begin{equation}
\begin{aligned}
\sum_{j\in N} x_{ij}^k - \sum_{j\in N} x_{ji}^k = \begin{cases} 
  b_k & \text{if } i = O(k) \\
  0 & \text{if } i \neq O(k), D(k) \\
  -b_k & \text{if } i = D(k)
\end{cases} \quad \text{for all } i \in N, k \in K
\end{aligned}
\end{equation}

\begin{equation}
x_{ij}^k \geq 0 \quad \text{for all } k \in K, (i,j) \in A
\end{equation}

\begin{equation}
y_{ij} \geq 0 \text{ and binary} \quad \text{for all } (i,j) \in A
\end{equation}

A comprehensive survey of network design problems is presented by Minoux[53] and Magnanti and Wong[47]. NDP can be classified as uncapacitated or capacitated. When there are no forcing constraints(2.42) imposed on any arcs, the problem is uncapacitated. NDP's have been studied extensively, however, optimal solutions even for relatively small problems are often
difficult to obtain. Optimal solution methods, including Branch-and-Bound, Benders decomposition, Lagrangian relaxation, dual-ascent and cutting plane algorithms, have been applied and the results are summarized in Balakrishnan[4], [5], Gendron[31] and Raghavan[64], etc. Dionne and Florian[27] suggest decomposition-based heuristics and approximation algorithms. Magnanti, Mireault and Wong[48] apply Benders decomposition and Balakrishnan, Magnanti and Wong[4] use a dual-ascent procedure with an add/drop heuristic. Los and Lardinois[45] evaluate a heuristic hill-climbing technique to solve uncapacitated network design problems. A hill-climbing algorithm starts from an initial solution and uses a systematic rule to improve on it until no improvement is possible by the rule, in order to achieve a local optimum. Balakrishnan[3] studies uncapacitated fixed-charge network design problems and shows that the path-flow formulation of the NDP is computationally attractive if the problem context imposes additional routing restrictions. For instance, long routes are often prohibited because of service constraints and other logistic considerations. Incorporating these restrictions as explicit constraints in an arc-flow formulation is difficult; therefore, the path-flow formulation is preferred, especially if the number of permissible routes is relatively small. Padberg, Van Roy and Wolsey[59] study zero-one mixed integer fixed-charge problems and develop two classes of facet-defining linear inequalities - knapsack inequalities and flow cover inequalities. Flow cover inequalities involve both flow and design variables, and have been used by Van Roy and Wolsey[71] in their cutting plane approach. Medhi[52] consider fixed-charge network design solution approaches to multi-hour combined capacity design and routing problems that arise in the design of dynamically reconfigurable broadband communication networks. Chang and Gavish[18] study the multiperiod network topology and capacity expansion problem in the telecommunication industry and propose tight lower bounding schemes within the framework of a Lagrangian relaxation.

Concerning transportation-related applications and solution methodologies, Billheimer and Gray[16] use link elimination/insertion heuristics to solve uncapacitated network design problems with an application to transit network design. Crainic and Rousseau[21] use decomposition heuristics and column generation principles for uncapacitated freight transportation service network design problems. Their decomposition-based algorithm heuristically solves the following two subproblems iteratively - service design and traffic routing. Lamar, Sheffi and Powell[41]
study uncapacitated, multicommodity network design problems in the less-than-truckload motor carrier industry and present a new lower bound for the problem and an efficient implementation scheme using shortest path and linearized knapsack programs.

Gendron and Crainic[31] analyze classical relaxation methods for a fixed charge multicommodity capacitated network design problem using resource-decomposition based solution methods. Magnanti, Mirchandani and Vachani[49], [51] consider capacitated network design problems in the telecommunication industry. They consider two approaches for solving the underlying mixed integer program: a Lagrangian relaxation strategy, and a cutting plane approach that adds cutset inequalities, arc residual capacity inequalities and 3-partition inequalities. A companion paper[50] shows that including these inequalities considerably improves LP performance and suggests that strong cutting planes can be an effective modeling and algorithmic tool for solving problems of the size that arise in the telecommunication industry. Balakrishnan, Magnanti and Mirchandani[5], [6] solve two-level network design problems that arise in the topological design of hierarchical communication, transportation, and electric power distribution networks. Balakrishnan, Magnanti and Wong[7] develop and test an optimization-based methodology to identify a minimum cost local access telecommunication network expansion plan to meet increasing demand. The solution method exploits the special tree and routing structure of the expansion planning problem to incorporate valid inequalities, obtained by studying the problem's polyhedral structure, within the context of a Lagrangian relaxation based dynamic program which solves an uncapacitated version of the problem. Bienstock and Gunluk[15] also study a version of the telecommunication capacity expansion problem. They study the polyhedral structure of a mixed-integer formulation of the problem with a cutting-plane algorithm using facet defining inequalities, which is very similar to Magnanti[51]. Clarke[19] study capacitated network design problems for telecommunication problems. The paper presents both link-based and path-based formulations and shows that the path-based model is more computationally effective than link-based models using branch-and-bound algorithms. They improve the LP performance by introducing valid inequalities developed by Magnanti, Mirchandani and Vachani[51] and propose SOS constraints to take advantage of SOS branching in MINOT (the Mixed INTEGRal Optimizer[56]).

Leung, Magnanti and Singhal[44] study a point-to-point route planning problem that arises
in many large scale delivery systems. The approach exploits the structure of the problem in order to decompose the problem into two smaller subproblems that are each amenable to solution by a combination of optimization and heuristic techniques based on Lagrangian relaxation. Farvolden and Powell[30] present local-improvement heuristics for a Service Network Design problem encountered in Less-Than-Truckload (LTL) common carrier applications. The add/drop heuristics are based upon subgradients derived from the optimal dual variables of the shipment routing subproblem. The basis of the multicommodity network flow problem is partitioned to facilitate the calculation of the dual variables, reduced costs and subgradients (see also Farvolden, Powell and Lustig[29]). Powell[62], Powell and Sheffi[63] also apply add/drop heuristics in LTL motor carrier applications. Most recently, Newton[58] studies network design problems under budget constraints with an application to railroad blocking problems.

2.6.3 Multicommodity Flow Problems

When the network is capacitated and the design variables are given, the NDP specializes to the multicommodity flow (MCF) problem. Assad[2], Kennington[39] and Ahuja, Magnanti and Orlin[1] present comprehensive surveys of literature, models and solution techniques for multicommodity network flow problems. The MCF problem is formulated as:

\[(MCF)\]

\[
\begin{align*}
\min & \sum_{k \in K} \sum_{(i,j) \in A} c_{ij}^k x_{ij}^k \\
\text{s.t. } & \sum_{k \in K} x_{ij}^k \leq u_{ij}, & (i,j) \in A \\
& \sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{ji}^k = \begin{cases} 
  b_k & \text{if } i = O(k) \\
  0 & \text{if } i \neq O(k), D(k) \\
  -b_k & \text{if } i = D(k) 
\end{cases}, & i \in N, k \in K
\end{align*}
\]
\[ x_{ij}^k \geq 0 \quad k \in K, \quad (i,j) \in A \]


### 2.6.4 Facility Location Problems

The facility location problem is another specialization of the NDP in which the design decisions concern the location and sizing of facilities at nodes of a distribution or transportation network. Van Roy[70] provides a comprehensive survey. Early work of Erlenkotter[28] on uncapacitated FLP's provides lower bounds from the dual problem that allow large combinatorial FLP problems to be solved. Guignard and Spielberg[32] show that inclusion of variable upper bounds give very tight lower bounds and sparse search trees. Van Roy[70] presents an implementation of the Cross Decomposition method to solve the capacitated facility location problem, which combines Benders decomposition and Lagrangian relaxation into a single framework that involves successive solutions to a transportation problem and an uncapacitated plant location problem. Guignard[33] uses Benders inequalities generated in a Lagrangian dual ascent procedure to solve uncapacitated plant location problems.
2.6.5 Vehicle Routing Problems

Consider a NDP with a single source, node $e$, and with a fixed capacity $u_{ij} = U$, on all arcs. In addition, suppose that the following assignment constraints on the design variables are imposed:

\[
\sum_{j \in N} y_{ij} = 1 \quad \text{for all } i \in N \setminus \{e\}
\]
\[
\sum_{i \in N} y_{ij} = 1 \quad \text{for all } j \in N \setminus \{e\}
\]

as well as the constraint

\[
\sum_{i \in N} y_{ej} \leq n.
\]

Then, the linear cost NDP becomes a vehicle routing problem (VRP) for a homogeneous fleet of $n$ vehicles, each domiciled at the source node (or depot) $e$ and each having capacity $U$. Comprehensive surveys by Magnanti[46], Magnanti and Wong[47] and the recent vehicle routing and scheduling survey by Desrosiers, et al.[26] and Desaulniers, et al.[24] summarize much of this field. A harder class of VRP's involve multiple depot, heterogeneous vehicle routing problems (MDHTVRP). Let $f$ be the set of vehicles and $E$ denote a set of depots ($e \in E$), then MDHTVRP can be formulated as:

(\text{MDHTVRP})

\[
\begin{align*}
\min & \sum_{f \in F} \sum_{(i,j) \in A} h_{ij}^f y_{ij}^f \\
\sum_{i \in K} x_{ij}^k & \leq \sum_{f \in F} u_f^j y_{ij}^f \quad \text{for all } (i, j) \in A
\end{align*}
\]

\[
\sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{ji}^k = \begin{cases} 
    b_k & \text{if } i = O(k) \\
    -b_k & \text{if } i = D(k) \\
    0 & \text{if } i \neq O(k), D(k)
\end{cases} \quad \text{for all } i \in N, k \in K
\]
\[
\sum_{j \in N} y_{ij}^f - \sum_{j \in N} y_{ji}^f = 0 \quad \text{for all } i \in N, \ f \in F \quad (2.49)
\]

\[
\sum_{f \in F} \sum_{j \in N} y_{ij}^f = 1 \quad \text{for all } i \in N \setminus E \quad (2.50)
\]

\[
\sum_{f \in F} \sum_{i \in N} y_{ij}^f = 1 \quad \text{for all } j \in N \setminus E \quad (2.51)
\]

\[
\sum_{j \in N} y_{ej}^f = \sum_{j \in N} y_{je}^f \leq n_e^f \quad \text{for all } e \in E \quad (2.52)
\]

\[
x_{ij}^k \geq 0 \quad \text{for all } k \in K, \ (i, j) \in A \quad (2.53)
\]

\[
y_{ij} \geq 0 \text{ and binary} \quad \text{for all } (i, j) \in A \quad (2.54)
\]

The objective function (2.46) is to find the cost minimizing set of vehicle routings, subject to constraints (2.47) - (2.54). Constraints (2.47), the \textit{forcing constraints}, limit the amount of flow on an arc to its capacity. Constraints (2.48), \textit{flow conservation constraints}, ensure that each commodity, from its origin to its destination, is fully serviced. Constraints (2.49) are the \textit{design balance constraints} that require the number of each vehicle type entering a node to equal the number leaving, for each node in the network. Constraints (2.50) and (2.51) assure that each demand is serviced exactly once given the assumption of \( b^k \leq \max_f (u_f) \) for all \( k \in K \). Constraints (2.52) require that each vehicle starting at a depot must return to that same depot. Finally, constraints (2.53) and (2.54) ensure nonnegativity of commodity flows and integrality and nonnegativity of design variables. Desrosiers[26] and Desaulniers, et al.[24] provide comprehensive surveys of time constrained routing and scheduling problems, including fixed schedule problems, traveling salesman problems, shortest path problems, vehicle routing problems, pickup and delivery problems, multicommodity network flow problems, bus driver scheduling problems and airline crew scheduling problems.

Ziarati, et al.[74] study the problem of assigning a sufficient number of locomotives of dif-
ferent types (heterogeneous consists) to trains to operate a pre-planned schedule. The original large-scale scheduling problem is decomposed and each smaller problem is then solved using a Dantzig-Wolfe decomposition method. For airline schedule planning and solution, Talluri and Gopalan[68] survey various mathematical models in airline schedule planning (i.e., fleet assignment, maintenance routing, traffic forecasting and allocation, equipment swapping, through flight selection and flight numbering) and their solution techniques. Shenvi[67] presents integrated airline schedule models (i.e., aircraft model and crew scheduling model) and solution methods. Aircraft model combines fleet assignment, through flight assignment and aircraft maintenance routing. Crew scheduling model integrates crew pairing and deadhead selection problems. Daskin and Panayotopoulos[23] study the problem of assigning aircraft to scheduled routes to maximize profits in passenger hub and spoke networks. A Lagrangian relaxation of the problem is outlined together with heuristics for converting the Lagrangian solutions into primal feasible solutions and for improving on the solutions.

Tables (2.2) and (2.3) summarize the network design literature. The third column in the tables represents whether the problem is modeled with the node-arc (link) formulation or the path formulation. In the last field of the tables, we report the size of the largest problem tested in the corresponding research. $N, A,$ and $K$ denote the number of nodes, links, and O/D commodities in the network, respectively. $Int, Cnt$ and $Row$ denote the number of integer variables, continuous variables, and constraints in the constraint matrix.
<table>
<thead>
<tr>
<th>Reference</th>
<th>Solution Algorithm</th>
<th>Model</th>
<th>Application / Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 46, A: 2,070)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(K: 1560)</td>
</tr>
<tr>
<td>Billheimer[16]</td>
<td>add/drop heuristic</td>
<td>link</td>
<td>transit network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 68, A: 476)</td>
</tr>
<tr>
<td>Chang[18]</td>
<td>Lagrangian relaxation, add/drop heuristic</td>
<td>link</td>
<td>telecommunication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 12, K: 40)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(Int: 396)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(Cnt: 15,840)</td>
</tr>
<tr>
<td>Crainic[21]</td>
<td>decomposition heuristic, column generation</td>
<td>path</td>
<td>rail</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 77, A: 681)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(Int: 399)</td>
</tr>
<tr>
<td>Dionne[27]</td>
<td>approximation algorithm</td>
<td>link</td>
<td>random network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 29, A: 54)</td>
</tr>
<tr>
<td>Lamar[41], [42]</td>
<td>Capacity Improvement (CI) based implicit enumeration</td>
<td>link</td>
<td>LTL motor carrier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 46, A: 2,070)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(K: 1560)</td>
</tr>
<tr>
<td>Loss[45]</td>
<td>hill-climbing, add/drop heuristic</td>
<td>link</td>
<td>random network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 30, A: 93)</td>
</tr>
<tr>
<td>Magnanti[48]</td>
<td>Benders decomposition</td>
<td>link</td>
<td>random network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 33, A: 100)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(K: 528)</td>
</tr>
<tr>
<td>Medhi[52]</td>
<td>genetic algorithm, Lagrangian relaxation</td>
<td>link</td>
<td>telecommunication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(N: 50, A: 82)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(K: 1225)</td>
</tr>
</tbody>
</table>

Table 2.2: Uncapacitated Network Design Applications
<table>
<thead>
<tr>
<th>Reference</th>
<th>Solution Algorithm</th>
<th>Model</th>
<th>Application / Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balakrishnan[5], [6]</td>
<td>dual-ascent, add/drop heuristic</td>
<td>link</td>
<td>random network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 500, A: 5,000 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 20,000 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Row: 5 \text{ million} )</td>
</tr>
<tr>
<td>Balakrishnan[7]</td>
<td>Lagrangian relaxation, dynamic programming</td>
<td>link</td>
<td>telecommunication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 41 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 2,870 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Row: 2,830 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 80,000 )</td>
</tr>
<tr>
<td>Bienstock[15]</td>
<td>cutting plane, valid inequalities</td>
<td>link</td>
<td>telecommunication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 16, A: 49 )</td>
</tr>
<tr>
<td>Clarke[19]</td>
<td>branch-and-price</td>
<td>path</td>
<td>telecommunication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 15, A: 32 )</td>
</tr>
<tr>
<td>Farvolden[30]</td>
<td>primal partitioning, add/drop heuristic</td>
<td>link</td>
<td>LTL motor carrier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 285, A: 1,065 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( K: 2,448 )</td>
</tr>
<tr>
<td>Gendron[31]</td>
<td>resource decomposition, valid inequalities</td>
<td>link</td>
<td>random network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 30, A: 520 )</td>
</tr>
<tr>
<td>Kuby[40]</td>
<td>branch-and-bound</td>
<td>path</td>
<td>package delivery</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 2,870 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Row: 2,830 )</td>
</tr>
<tr>
<td>Leung[44]</td>
<td>Lagrangian relaxation</td>
<td>link</td>
<td>random network</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 250, A: 2,000 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( K: 8,000 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 16 \text{ million} )</td>
</tr>
<tr>
<td>Magnanti[51]</td>
<td>Lagrangian relaxation, cutting plane</td>
<td>link</td>
<td>telecommunication</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 15, A: 34 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( K: 21 )</td>
</tr>
<tr>
<td>Newton[58]</td>
<td>branch-and-price</td>
<td>path</td>
<td>railroad blocking</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 6,802 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Row: 43,186 )</td>
</tr>
<tr>
<td>Pandit[60]</td>
<td>branch-and-cut</td>
<td>link</td>
<td>air-cargo distrib.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Int: 3,830 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Cnt: 59,124 )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( Row: 9,265 )</td>
</tr>
<tr>
<td>Powell[62], [63]</td>
<td>add/drop heuristic</td>
<td>link</td>
<td>LTL motor carrier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( N: 300 )</td>
</tr>
</tbody>
</table>

Table 2.3: Capacitated Network Design Applications
Chapter 3

A Multimodal Express Package Shipment Operation

3.1 Problem Description

This chapter describes the service network design problem of a large provider of express package service. The objective is to find the cost minimizing movement of packages from their origins to their destinations using a limited number of resources, such that service commitments are satisfied. There are multiple products or service types, defined by the speed of service required. Service occurring overnight, the most costly service, is referred to as Next-Day Service, while guaranteed service within 48 hours is called Second-Day Service and service within 3-5 days is called Deferred Service.

This service network design problem, like others, can be thought of as two major decisions: the first is to determine the service network and the second is to determine the flows over the service network. The service network is defined by the movements in time and space of the transportation assets, in this case, jets, propeller aircraft, and ground vehicles. The specific package routings from origins to destinations determine the flows on each movement or link in the service network.
3.1.1 Service Network and Package Flows

Shipments originate and terminate at airport locations, called gateways, where packages are loaded, unloaded and transferred between different jets, propeller aircraft and/or ground vehicles. From its initial gateway, a package may be flown to at most one additional gateway before it is flown to a hub. Hubs are specialized gateways that perform the additional function of sorting packages. At the hub, a package is unloaded from an inbound (into the hub) jet, sorted and loaded on an outbound (out of the hub) jet. Again, it may be flown to at most one intermediate gateway before being flown to its final destination. In Fig.3-1, a representative service network and routing for one package is depicted.

Packages are transported from origin to destination using the following modes:

1. Exclusive feeder transportation. Exclusive feeder transportation involves the transport of packages from their origins to their destinations using only ground vehicles and/or propeller aircraft. Since ground vehicle and propeller aircraft transport are cheap compared to jets, a package will always use exclusive feeder transportation if service requirements can be satisfied. As a result, any package that qualifies is assigned to exclusive feeder service and is removed from further consideration in our model.

2. Combined transportation. Typically, service requirements cannot be satisfied using exclusively the slower propeller aircraft and ground vehicle modes. Instead, transport of packages from origins to destinations is achieved using a combination of jet, propeller aircraft and ground vehicle transport.

Associated with gateways are earliest pickup times (EPT's) and latest delivery times (LDT's) that capture service considerations. An EPT denotes the time at which packages will be available for pickup at a gateway. Each gateway's EPT is scheduled as late as possible to allow customers time to prepare their shipments, but early enough so that delivery service standards can be met. A gateway's LDT denotes the time by which all packages must be delivered to a location in order to satisfy delivery standards. In setting EPT's and LDT's, we also consider hub sort capacities and time windows designating the start and end sort times.

An aircraft route can be decomposed into two distinct components, a pickup route and a delivery route. A pickup route typically departs from some gateway in the early evening.
Figure 3-1: Express Package Shipment Operation
and is restricted to contain at most one intermediate stop before its final stop at a hub. A delivery route begins at a hub, typically departing in the early morning, and stops at most at two gateways. The number of stops on a pickup or delivery route is restricted to three to limit the potential of schedule problems arising in this hub-and-spoke network. Fewer take-offs and landings result in a reduction in the expected schedule slippage. Given the time sensitive nature of the express package delivery operation, robustness of operation is critical.

A pickup or delivery route is feasible only if all service requirements can be fully satisfied. For example, in Fig.3-1, we have two aircraft pickup routes, one is \((1 \rightarrow 2 \rightarrow H)\) by aircraft type 1 and the other is \((2 \rightarrow H)\) by aircraft type 2. Numbers beside aircraft legs denote the transportation time between two locations. Numbers in the brackets denote the earliest pickup time (in case of pickup routes) or latest delivery time (in case of delivery routes) at gateways. At hubs, sort start times and sort end times are specified. Pickup route \((1 \rightarrow 2 \rightarrow H)\) by aircraft type 1 is feasible because the earliest arrival time \((EAT)\) at the hub \([8:00\ P.M.\ (EPT)\ of\ gateway\ 1)\ + \ 4.5\ hour\ (travel\ time) = 12:30\ A.M.]\ is before the hub sort end time of 4:00 A.M.. Likewise, pickup route \((2 \rightarrow H)\) by aircraft type 2 is feasible. If the sort end time were 2:00 A.M. instead, the pickup route \((2 \rightarrow H)\) by aircraft type 2 would not be feasible because its earliest arrival time to the hub is too late. Pickup route \((1 \rightarrow 2 \rightarrow H)\) by aircraft type 1, however, is still feasible. The delivery route \((H \rightarrow 3 \rightarrow 1)\) is feasible because the EAT at gateway 3 (sort start time of H at 1:00 A.M. + 3 hour travel time = 4:00A.M.) is earlier than its latest delivery time \((LDT)\) of 9:00 A.M. Again, if the \(LDT\) of gateway 3 were 3:00 A.M. instead, then this delivery route would not be feasible.

Since shipments can be transferred between aircraft at gateways, the number of shipment routes far exceeds the number of aircraft routes. Fig.3-1 illustrates this fact. There are three possible shipment routes and only two aircraft routes.

### 3.1.2 Costs

The costs of an express package operation can be expressed as the sum of jet aircraft costs, feeder route costs and package handling costs. For a given flight leg and aircraft type, jet aircraft operating costs are the sum of associated fuel cost, crew cost, cycle cost (that is, the cost of take-off and landing), maintenance cost, etc. Depending on the nature of the model, jet
ownership cost may or may not be included in the model. Ownership cost is not included for near-term planning because the company already owns the aircraft, however, a fixed cost per aircraft is included if the model is to be used in a more strategic nature, e.g., if the model is to be used to determine future fleet composition. The cost for ground movements and feeder air movements is expressed as the total cost for a given leg, for each fleet type. Package handling costs are expressed per package for each gateway and hub location.

3.2 The Planning Problem

For most express package service operations, there are at least two types of planning activities, largely characterized by their planning time horizon. One, termed strategic long range planning looks several years into the future. This type of planning is focused on problems of aircraft acquisition, hub capacity expansion, new facility location, etc. The planners are not constrained by existing resources. As a matter of fact, determining the required resources under the future operating conditions is one of the objectives of this type of planning activity. Although the data used in this type of planning is often imprecise, relying heavily on forecasts, the planners must construct an operating plan to assess various scenarios.

The second type of planning is near-term operation planning. This planning activity, considering anything from one to several months in the future, generates a plan to be executed in the operation. Other downstream planning activities, such as flight crew planning and maintenance planning, are based on the output of this process. For obvious reasons, in near-term planning, there are very limited degrees of freedom in terms of changing existing resources. However, it is common to use these models to analyze various scenarios, like determining the incremental operating costs for a set of volume changes. The results are used to direct marketing efforts over the next one to two years. We call this type of analysis market planning.

While our primary objective is to facilitate strategic long range planning, we design our models and algorithms to be applicable in the contexts of near-term operation planning and market planning.

In the case of the company we consider, planning is still done primarily manually, with limited machine intervention. In the case of near-term planning, a database system is used to
generate reports to compare operations and plan, but no automated decision support per se is available. As a result, it takes an entire year to develop a Long Range plan. The models we develop will shorten this planning cycle and allow extensive scenario analyses to be performed.

3.2.1 Problem Scope

Our ultimate goal is to simultaneously solve the aircraft routing/scheduling/package flow problem over a seven day planning horizon for all products (i.e., next day, second day, deferred). We determine both the service network and the package flows over this network. The output, then, specifies the schedule of every feeder and jet route, the vehicle or aircraft type assigned to each route, and the package routings in the resulting network.

The model inputs include package movement requirements, a network of potential feeder and jet movements (called legs), fleet composition and characteristics, and sort capabilities and characteristics. The fleet composition is given, with an option available to lease additional aircraft. All aircraft, gateway and hub operating characteristics, such as range, speed, runway length, sort capacity, etc., are considered fixed.

To capture the characteristic daily variations in express package operations, that is, volumes increase from Monday through mid-week, peaking on Wednesday or Thursday, and decrease through the weekend, a seven day planning horizon is optimal. A seven day horizon has the added advantage that all service types, e.g., next day, second day and deferred, can be considered. The drawback, however, is that a week-long planning horizon considering all service types has associated with it a huge amount of data and a huge (likely intractable) model. As a result, in this research we scale back our ultimate objective and focus on the goal of solving only the next day problem over a single day planning horizon. We break this problem into two phases (see Figure 3-2), summarized as follows.

Phase 1: Package Flow Model and Solution

In this phase, a single day planning horizon is considered, only next day packages are considered, the vehicle and aircraft routes and schedules are fixed, and the only decision variables involve the routing of packages. This problem can be cast as a multicommodity network flow problem, with all packages with a common origin and destination specified as a single commod-
ity. The number of commodities for our application ranges from 8,000 to 200,000, presenting a challenge to standard models and solvers for multicommodity flow problems. It is essential that this problem be solved quickly since it is an embedded problem within the Phase 2 problem.

Phase 2: Simultaneous Service Network Design and Package Flow Model and Solution

In this phase, we simultaneously determine the jet and feeder routes and schedules, and the next day package flows over a single day planning horizon. This problem is a network design problem of immense proportions.

Although we focus only on a short planning horizon and only one type of product, our phased approach allows us to:

- Assess the solvability of the ultimate multi-product, week-long planning horizon problem.
- Develop modeling techniques and solution methodologies for the ultimate problem by solving smaller self-contained problems, that by themselves are important.
Chapter 4

The Service Network

4.1 Network Representation

In time constrained routing and scheduling problems, the conventional approach is to build a time-space network, as defined in Chapter 2. For this application, each node in the time-space network corresponds to the origin or destination of a jet or feeder movement at some point in time and each arc corresponds to a jet or feeder movement at a particular time. The drawback to this time-space representation is that it leads to an enormous network. So, we exploit the structure of our express package delivery problem, namely, the fact that at most three locations can be visited on a pickup or delivery route, and represent all movements and feasible package routings using a network of drastically reduced size. We refer to this mechanism for reducing network size as the derived schedule approach since it exploits the fact that we don't need to know the exact schedule for each move, we need only to know that there exists a feasible schedule.

For each jet type, we generate its derived schedule network (see Figure 4-1 and 4-2). For now, consider one fleet type, i.e., a particular type of jet, propeller aircraft or ground vehicle (we omit the notation designating fleet type for ease of exposition.) Each pickup route consists of at most two legs: the first from a starting gateway to either a hub or an intermediate gateway, and the second, if it exists, from an intermediate gateway to a hub. Similarly, each delivery route consists of at most two legs: the first from a hub to a gateway, and the second, if it exists, from this intermediate gateway to another gateway. In our derived schedule network for
Figure 4-1: Derived Pickup Network

Figure 4-2: Derived Delivery Network
a single fleet, we represent routes using the following node and arc sets:

\[ N^P_1 (N^D_1) \subset N : \text{gateway nodes representing first (last) stops on pickup (delivery) routes.} \]

\[ N^P_2 (N^D_2) \subset N : \text{pair nodes representing intermediate stops on pickup (delivery) routes.} \]

\[ H : \text{hub nodes representing last (first) stops on pickup (delivery) routes.} \]

\[ L^P (L^D) \subset A : \text{leg arcs representing legs in pickup (delivery) routes.} \]

\[ G^P (G^D) \subset N : \text{ground arcs representing periods during which aircraft/vehicle location does not change in pickup (delivery) routes.} \]

The sets \( N^P_1 \) and \( N^D_1 \) each contain one node for each gateway. For ease of exposition, we let gateway node \( i \in N^P_1 \cup N^D_1 \) correspond to gateway \( i \). \( H \) contains one node for each hub, and similarly, hub node \( h \) corresponds to hub \( h \). \( N^P_2 \) and \( N^D_2 \) each contain one node for each ordered pair of gateways and each ordered pair consisting first of a gateway and second, a hub, for a total of \(|G|^2 + |G||H|\) nodes where \(|G|\) is the number of gateways and \(|H|\) is the number of hubs.

Each gateway node \( i \in N^P_1 \) has an associated time \( t_i = EPT(i) + \text{time to load packages at } i \), representing the earliest time that an aircraft/vehicle may begin its pickup route at gateway \( i \). Similarly, for each \( i \in N^D_1 \), \( t_i = LDT(i) - \text{time to unload packages at } i \), reflects the latest time that an aircraft/vehicle may end its delivery route at gateway \( i \) so that service requirements may be satisfied. Each hub node \( h \in H \) for hub \( h \) has an associated time \( t_h = SET(h) \), the sort end time at \( h \). All pickup routes must arrive at \( h \) before \( t_h - \text{the time to unload packages at } h \), and all delivery routes may not depart \( h \) before \( t_h + \text{the time to load packages at } h \). The time \( t_i \) of the pair node \( i \in N^P_2 \) for gateways \( m \) and \( n \), is the earliest time that an aircraft/vehicle initiating its pickup route at \( m \) can depart its intermediate stop at \( n \), i.e., \( t_i = \max[EPT(n), t_m + \text{travel time}(m \to n)] \). If \( m = n \), then \( t_i = t_m \). Similarly, the time \( t_i \) of the pair node \( i \in N^P_2 \) for gateway \( n \) and hub \( h \), is the latest time that an aircraft/vehicle on its pickup route may depart \( n \) in order to arrive at \( h \) before its sort end time, i.e., \( t_i = \max[EPT(n), t_h - \text{travel time}(n \to h) - \text{time to unload packages at } h] \). With regard to delivery routes, the time \( t_i \) of the pair node \( i \in N^D_2 \) for hub \( h \) and gateway \( n \), is the earliest time that an aircraft/vehicle initiating its delivery route at \( h \) may arrive at the intermediate stop at \( n \), i.e., \( t_i = \min[LDT(n), t_h + \text{travel time}(h \to n) + \text{time to load packages at } h] \). Finally, the time \( t_i \) of the pair node \( i \in N^D_2 \) for gateway \( n \) and gateway \( m \), is the latest time that an aircraft/vehicle may leave \( n \).
on its delivery route to \( m \) in order to satisfy service requirements at \( m \), i.e., \( t_i = t_m - \text{travel time}(n \to m) \). Again, if \( n = m \), then \( t_i = t_m \).

We construct the set of leg arcs for pickup routes as follows:

- For each \( i \in N_1^P \) and \( l \in N_2^P \), we include an arc from gateway node \( i \) to a pair node \( l \) for the ordered pair \((i, j)\), for any gateway \( j \) as long as the travel time \((i \to j)\) does not exceed the maximum time allowed.

- For each \( l \in N_2^P \) and \( h \in H \), we include an arc to hub node \( h \) from pair node \( l \) for the ordered pair \((i, h)\), for any gateway \( i \), \( i \neq h \), if the travel time \((i \to h)\) does not exceed the maximum time allowed and \([t_h - \text{travel time}(i \to h) - \text{time to unload packages at } h] \geq t_i\). (If this inequality is not satisfied, it is not feasible to service gateway \( n \) through hub \( h \).)

Similarly, for delivery routes, we build the set of leg arcs as follows:

- For each \( l \in N_2^D \) and \( h \in H \), we include an arc from hub node \( h \) to pair node \( l \) for the ordered pair \((h, j)\), for any gateway \( j \), \( h \neq j \), if the travel time \((h \to j)\) does not exceed the maximum time allowed and \( t_l \geq t_h + \text{travel time}(h \to j) + \text{time to load packages at } h \). (If this inequality is not satisfied, it is not feasible to service gateway \( j \) through hub \( h \).)

- For each \( i \in N_1^D \) and \( l \in N_2^D \), we include an arc to gateway node \( i \) from pair node \( l \) for the ordered pair \((i, j)\), for all gateways \( j \), if the travel time \((j \to i)\) does not exceed the maximum time allowed.

Finally, we construct the set of ground arcs in the derived schedule network as follows:

- We assign gateway node \( i \in N_1^P(N_1^D) \) to gateway \( i \), pair node \( l \in N_2^P \) for the ordered pair \((i, j)\) of gateways to gateway \( j \), pair node \( l \in N_2^P \) for the ordered pair \((j, h)\) for gateway \( j \) and hub \( h \) to gateway \( j \), hub node \( h \in H \) to hub \( h \), pair node \( l \in N_2^D \) for the ordered pair \((i, j)\) of gateways to gateway \( i \), and pair node \( l \in N_2^D \) for the ordered pair \((h, j)\) for hub \( h \) and gateway \( j \) to \( j \).

- We sort all nodes assigned to gateway \( j \) in increasing order of time and place a ground arc between each pair of successive nodes, for each \( j \). The from node of any ground arc has time not later than its to node.
The cost of each leg arc consists of fleet routing cost and package routing cost, and fleet routing cost is composed of fixed cost and operation cost. Each fleet type has different fixed cost and variable operating cost structure and variable operating cost is calculated as \([\text{unit operating cost ($/time)} \times \text{travel time}]\). Package routing cost is expressed as unit package handling cost per each leg arc, that is, \([\text{unit package handling cost ($/package/time)} \times \text{travel time}]\).

The capacity of each leg arc corresponds to the capacity of fleet type used. All cost elements for each ground arc are set to zero, and the capacity of each ground arc is unlimited.

We construct one derived schedule network for propeller aircraft, one for ground vehicles and one for each type of jet, since times associated with gateway and pair nodes vary for fleet type. By design, the derived schedule network for any fleet type \(f\) has the following characteristics:

- Each network path beginning at a gateway node and ending at a hub node corresponds to a feasible pickup route for fleet type \(f\).

- Each network path beginning at a hub node and ending at a gateway node corresponds to a feasible delivery route for fleet type \(f\).

- Each feasible pickup or delivery route for fleet type \(f\) is represented by a path in the network. Pickup (delivery) routes that contain only one leg, e.g., a pickup route from gateway \(i\) to hub \(h\) are represented by a path containing the leg arc from gateway node \(i\) to pair node \(l\) for the pair \((i, i)\) of gateways, the ground arc from pair node \(l\) to pair node \(m\) for the pair \((i, h)\) representing gateway \(i\) and hub \(h\), and the leg arc from pair node \(m\) to hub node \(h\).

- Each route has potentially many feasible schedules. To illustrate, consider a pickup route \((i - j - h)\) represented by gateway node \(i\), pair node \(m\), pair node \(n\), and hub node \(h\). This route can begin as early as \(t_i\) and as late as \(t_i + (t_n - t_m)\), where \((t_n - t_m)\) is nonnegative for each feasible route. If the route start time is \(t_i + \partial\), for \(\partial \leq (t_n - t_m)\), then the time of departure from intermediate gateway \(j\) is \((t_m + \partial + \gamma)\), where \(\gamma \leq (t_n - t_m - \partial)\). Finally, the arrival at \(h\) will be \(t_m + \partial + \gamma + \text{travel time} (j - h)\).

We represent the set of potential routes and schedules for packages using a package network,
constructed by merging the derived schedule network for each fleet type (see Figure 4-3 and 4-4). The package network is constructed as follows:

- Merge gateway and hub nodes from the derived schedule networks for each fleet:
  
  1. For each fleet \( f \in F \), map its gateway node \( i \), for all \( i \in N_i^F \) (\( i \in N_i^P \)), into a single gateway node \( i \in N_i^P \) in the package network.
  
  2. For each fleet \( f \in F \), map its hub node \( h \), for all \( h \in H \), into a single hub node \( h \in H \) in the package network.

- Add to the package network all leg arcs and pair nodes in the derived schedule network for fleet type \( f \), for all \( f \in F \).

- Sort all pair nodes at a single location in increasing order of time, and add ground arcs between successive pair nodes at a location, for each location.

Arc costs and capacities in the package network are as defined earlier.

Like the derived schedule network, the package network has the following characteristics:

- Each network path beginning at any gateway node \( i \), containing a hub node and ending at any gateway node \( j \) is a feasible route for packages originating at \( i \) and destined for \( j \).

- Each feasible route for any package originating at \( i \) and destined for \( j \) is represented by a path in the package network.

- Each route has potentially many feasible schedules.

Figure 4-3 and 4-4 are the network which can generate both fleet routes and package flows. Let's say that the solid line represents fleet type 1 and dotted line is for fleet type 2 and there is a service demand from gateway 2 to gateway 3. Fleet type 2 has slower operating speed than fleet type 1. In order to service the demand, it is not necessary that fleet routes must start from 2 and end at 3. It can also be serviced by any fleet routes that visit 1 and 3 as intermediate gateways. Since the number of possible routes are too many to enumerate, let's make another assumption, without loss of generality, that a package demand must go through hub 1. Table
Figure 4-3: Derived Pickup Package Network with Two Fleet Types

Figure 4-4: Derived Delivery Package Network with Two Fleet Types
<table>
<thead>
<tr>
<th></th>
<th>Pickup Fleet Routes</th>
<th>Delivery Fleet Routes</th>
</tr>
</thead>
<tbody>
<tr>
<td>fleet type 1</td>
<td>1 → 2 → h1</td>
<td>h1 → 1 → 3</td>
</tr>
<tr>
<td></td>
<td>2 → h1</td>
<td>h1 → 2 → 3</td>
</tr>
<tr>
<td></td>
<td>2 → 1 → h1</td>
<td>h1 → 3</td>
</tr>
<tr>
<td></td>
<td>2 → 3 → h1</td>
<td>h1 → 3 → 1</td>
</tr>
<tr>
<td></td>
<td>3 → 2 → h1</td>
<td>h1 → 3 → 2</td>
</tr>
<tr>
<td>fleet type 2</td>
<td>1 → 2 → h1</td>
<td>h1 → 2 → 3</td>
</tr>
<tr>
<td></td>
<td>2 → h1</td>
<td>h1 → 3</td>
</tr>
<tr>
<td></td>
<td>2 → 1 → h1</td>
<td>h1 → 3 → 1</td>
</tr>
<tr>
<td></td>
<td>3 → 2 → h1</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4.1: Possible Pickup and Delivery Fleet Routes

4.1 shows all possible pickup and delivery fleet routes which can service 1 → 3 demand through h1.

Note that pickup fleet route 2 → 3 → h1 and delivery fleet routes h1 → 1 → 3 and h1 → 3 → 2 are infeasible for fleet type two because $t_n > t_m$ for pair node m, pair node n, and hub node h. Since shipment can be transferred between aircraft at gateways, the number of shipment routes far exceeds the number of aircraft routes. For example, with regard with 2 → 1 → h1 → 3 movement, all possible shipment routes can be enumerated as:

- $2 \rightarrow 1 \rightarrow h1 \rightarrow 3$
- $\begin{array}{c}
    \text{fleet type 1} \\
    \text{fleet type 1}
\end{array}
\rightarrow h1 
\begin{array}{c}
    \text{fleet type 1} \\
    \text{fleet type 1}
\end{array}
\rightarrow 3$
- $\begin{array}{c}
    \text{fleet type 1} \\
    \text{fleet type 2}
\end{array}
\rightarrow h1 
\begin{array}{c}
    \text{fleet type 1} \\
    \text{fleet type 2}
\end{array}
\rightarrow 3$
- $\begin{array}{c}
    \text{fleet type 1} \\
    \text{fleet type 2}
\end{array}
\rightarrow h1 
\begin{array}{c}
    \text{fleet type 1} \\
    \text{fleet type 2}
\end{array}
\rightarrow 3$
- $\begin{array}{c}
    \text{fleet type 2} \\
    \text{fleet type 2}
\end{array}
\rightarrow h1 
\begin{array}{c}
    \text{fleet type 2} \\
    \text{fleet type 2}
\end{array}
\rightarrow 3$
- $\begin{array}{c}
    \text{fleet type 2} \\
    \text{fleet type 2}
\end{array}
\rightarrow h1 
\begin{array}{c}
    \text{fleet type 2} \\
    \text{fleet type 2}
\end{array}
\rightarrow 3$
- $\begin{array}{c}
    \text{fleet type 2} \\
    \text{fleet type 2}
\end{array}
\rightarrow h1 
\begin{array}{c}
    \text{fleet type 2} \\
    \text{fleet type 2}
\end{array}
\rightarrow 3$
4.2 Network Reduction Methods

Although our derived schedule service network avoids the massive explosion in the number of nodes and links of a conventional time-space network, the network is still very large. Specifically, one data set describing our particular application has 75,944 nodes and 151,002 links. We reduce the size of this network dramatically by using two network reduction methods, namely, node consolidation (as described in Chapter 2) and link consolidation.

When we apply node consolidation to our network of 75,944 nodes and 151,002 links, we achieve a reduced-size network containing 2,400 nodes and 72,144 links. This dramatic decrease in size can be explained by examining the particular structure of our network. Each pair node is associated with either the arrival or the departure of a leg arc. And, pair nodes for arrivals typically have associated time earlier than pair nodes for departures. The result is massive node consolidation and corresponding link elimination. For example, when we apply node consolidation to Figure 4-3, we reduce the network with 24 nodes and 21 ground arcs to 5 nodes and 2 ground arcs. Figure 4-5 shows the result.

After node-consolidation, there are many instances of parallel arcs between two nodes in the network. These arcs represent movements of different fleets between locations at roughly
the same time. We will show in the Chapter 6 that it is advantageous from the point of view of the formulation to represent these replicate arcs by a single arc, recording the set of relevant characteristics of the various equipment types that can be assigned to the corresponding leg. To illustrate this link consolidation, consider Figure 4-5 containing a network representation before link consolidation and contrast it with Figure 4-6 representing the same network after link consolidation: 24 leg arcs are reduce to 15. Applying link consolidation to our network of 72,144 links (after node consolidation), we reduced the number of links to 15,355. In total, our combined node and link consolidation allowed us to reduce network size from 75,944 nodes and 151,002 links to 2,400 nodes and 15,355 links.
Chapter 5

Package Flow Models and Solutions

Given a service network, the objective of the package flow problem is to find the minimum cost flow of packages from their origins to their destinations satisfying service commitments and network capacity. Alternatively, for a given service network, the objective may be to determine whether a set of package flows can be serviced and if not, to determine where the network lacks sufficient capacity. Yet another objective may be to determine if service standards can still be met if they are changed, for some or all packages. Whichever objective, these problems all can be cast as linear multicmodity network flow problems. Multicommodity flow (MCF) problems have been studied extensively, with a survey of linear multicommodity flow problems in Ahuja et al.[1], and discussion of a modeling and solution approach for integer multicommodity flow problems (i.e., every commodity must be assigned to exactly one path from its origin to its destination) in Barnhart et al.[14].

5.1 Three MCF Formulations: Node-Arc, Path and Tree

There are several ways to formulate a MCF problem, with the best formulation dependent on the particular problem characteristics. The three formulations we consider are the node-arc formulation, the path formulation and the tree formulation. We describe and contrast these formulations and discuss their solutions next.
5.1.1 Node-Arc Formulation

The node-arc formulation is perhaps the most known formulation of the MCF problem (see Assad[2], Kennington[39], Helgason, Kennington and Lall[35], Barnhart and Shefi[10]). To facilitate our description of the node-arc formulation for the MCF problem we introduce the following notations.

SETS

\( N \) : node set
\( A \) : arc set
\( K (\ni k) \) : Set of commodities, with each \( k \in K \) defined by an origin \( O(k) \) and a destination \( D(k) \)

PARAMETERS

\( c_{ij}^k \) : cost of sending one unit of commodity \( k \in K \) on arc \( (i,j) \in A \)
\( b^k \) : total quantity of commodity \( k \in K \)
\( u_{ij} \) : capacity of arc \( (i,j) \in A \)

DECISION VARIABLES

\( x_{ij}^k \) : fraction of \( b^k \) on arc \( ij \in A \), for \( k \in K \)

The resulting node-arc formulation is:

\[
\begin{align*}
\text{min} & \quad \sum_{(i,j) \in A} \sum_{k \in K} c_{ij}^k b^k x_{ij}^k \\
\sum_{j \in N} x_{ij}^k - \sum_{j \in N} x_{ji}^k & = \begin{cases} 
1 & \text{if } i = O(k) \\
-1 & \text{if } i = D(k) \\
0 & \text{if } i \neq O(k), D(k)
\end{cases} \quad \text{for all } i \in N, k \in K \\
\sum_{k \in K} b^k x_{ij}^k & \leq u_{ij} \quad \text{for all } (i,j) \in A \\
x_{ij}^k & \geq 0 \quad \text{for all } k \in K, (i,j) \in A
\end{align*}
\]
The objective function (5.1) is to minimize the total flow cost of commodities from their origins to their destinations. Constraints (5.2), imposed on each commodity \( k \in K \), guarantee that flow is conserved from \( O(k) \) to \( D(k) \). Constraints (5.3), called "bundle constraints", guarantee that the total flow on arc \((i,j)\) does not exceed capacity of \((i,j)\), for all \((i,j) \in A\). The constraints (5.4) ensure that flow is nonnegative on every arc \((i,j) \in A\).

### 5.1.2 Node-Arc Solution

The node-arc formulation for the MCF problem can be solved directly using any linear program (LP) solver, in theory. This issue in practice is that the size of the node-arc LP may be prohibitive in the sense that its solution may require excessive amounts of memory. The node-arc formulation contains \(|A| \times |K|\) variables and \(|N| \times |K| + |A|\) constraints, where \(|S|\) denotes the size of the set \( S \). For problems with a large network and/or a large number of commodities, direct solution of the node-arc LP may be impractical.

### 5.1.3 Path Formulation

To overcome the size difficulties associated with the node-arc MCF formulation, we consider the path formulation of the MCF problem. This formulation was first presented by Tomlin[69] and has since been applied extensively by Barnhart[9], Barnhart et al.[13], Farvolden, Powell and Lustig[29], Jones et al.[38], Sheffi [66], etc.

Before presenting the formulation, we first define additional notations.

**SETS**

\( P^k (\ni p) : \) set of paths from \( O(k) \) to \( D(k) \) for each \( k \in K \)

**PARAMETERS**

\( c^*_p \): cost of sending one unit of flow of \( k \in K \) along path \( p \in P^k \)

**INDICATOR VARIABLES**

\( \delta^*_{ij} : \begin{cases} 1 & \text{if arc } (i,j) \in A \text{ belongs to } p \\ 0 & \text{otherwise} \end{cases} \), where \( p \in P^k, k \in K \)

**DECISION VARIABLES**

\( x^k_p : \) fraction of \( b^k \) on path \( p \in P^k \) for all \( k \in K \)
The path formulation for the MCF problem is as follows.

\[
\min \sum_{k \in K} \sum_{p \in P^k} \left( c_p^{k,k} \right) x_p^k
\]  
(5.5)

\[
\sum_{p \in P^k} x_p^k = 1 \quad \text{for all } k \in K
\]  
(5.6)

\[
\sum_{k \in K} \sum_{p \in P^k} \left( \delta_p^{i,j} b^k \right) x_p^k \leq u_{ij} \quad \text{for all } (i, j) \in A
\]  
(5.7)

\[
x_p^k \geq 0 \quad \text{for all } k \in K, p \in P^k
\]  
(5.8)

Again, the objective is to minimize the total cost of flowing commodities from their origins to their destinations. Constraints (5.6), generalized upper bounding constraints, ensure that the total quantity of each commodity is assigned to the network. Constraints (5.7) guarantee that the total arc flow does not exceed arc capacity, for each \((i, j) \in A\) and constraints (5.8) ensure that path flows are all nonnegative.

### 5.1.4 Path Solution

Compared with the node-arc formulation, the path formulation has a reduced number of constraints from \(|N| \times |K| + |A|\) to \(|K| + |A|\), a significant reduction. The number of variables, however, has increased exponentially from \(|A| \times |K|\) to \(|\Pi|\), where \(|\Pi|\) represents the number of paths for all commodities \(k \in K\). If direct solution of the node-arc formulation is impractical, then clearly so is direct solution of the path formulation. Column generation methods, however, allow the path formulation to be solved indirectly. That is, as described in Chapter 2, rather than solving the huge LP once, column generation methods solve the path formulation by repeatedly solving much smaller restricted problems. The restricted problems contain all constraints, but only a subset of the variables. The variable set is augmented by solving a pricing problem that selects variables that may improve the objective function value. The pricing problem for the path formulation requires the solution of the following mathematical program:
\[ z^* = \min_{p \in P^k, k \in K} \sum_{(i,j) \in A} \left( \delta^p_{ij} b^k \right) \pi_{ij} - \sum_{(i,j) \in A} \left( \delta^p_{ij} b^k \right) \pi_{ij} - \sigma_k \]

where \( \pi \) are non-positive and \((\sigma, \pi)\) are the dual variables associated with constraints (5.6) and (5.7), respectively. The reduced cost of path \( p \) given solution \((\sigma, \pi)\) is represented by \( c^p_k b^k - \sum_{(i,j) \in A} \left( \delta^p_{ij} b^k \right) \pi_{ij} - \sigma_k \). As a result, if \( z^* \geq 0 \) then the path formulation LP is solved. However, if \( z^* < 0 \), the solution to the pricing problem identifies a path for inclusion in the restricted LP (i.e., a variable or column is generated).

To solve the pricing problem, notice that the problem can be rewritten as:

\[
\begin{align*}
z^* &= \min_{p \in P^k, k \in K} c^p_k b^k - \sum_{(i,j) \in A} \left( \delta^p_{ij} b^k \right) \pi_{ij} - \sigma_k \\
&= \min_{p \in P^k, k \in K} \sum_{(i,j) \in A} c^k_{ij} b^k \delta^p_{ij} - \sum_{(i,j) \in A} (\pi_{ij} b^k) \delta^p_{ij} - \sigma_k \\
&= \min_{p \in P^k, k \in K} b^k \sum_{(i,j) \in A} \left( c^k_{ij} - \pi_{ij} \right) \delta^p_{ij} - \sigma_k
\end{align*}
\]

If \( b^k \sum_{(i,j) \in A} \left( c^k_{ij} - \pi_{ij} \right) \delta^p_{ij} \geq \sigma_k, \forall p \in P^k \), for every \( k \in K \), then each path (or variable) has non-negative reduced cost and the path formulation is solved. Equivalently, if \( \min_{p \in P^k, k \in K} b^k \sum_{(i,j) \in A} \left( c^k_{ij} - \pi_{ij} \right) \delta^p_{ij} \geq \sigma_k \), for every \( k \in K \), optimality is achieved. The problem of determining the path that minimizes \( b^k \sum_{(i,j) \in A} \left( c^k_{ij} - \pi_{ij} \right) \delta^p_{ij} \) is a shortest path problem for each \( k \in K \) over the service network with modified link costs of \( c^k_{ij} - \pi_{ij} \), for all \((i,j) \in A\). This implies that the pricing problem for the path formulation is solved by finding the shortest path for each \( k \in K \). Efficient solution of the pricing problem is important, especially for applications where it may be solved several hundreds of thousand times.

### 5.1.5 Tree Formulation

Even the path formulation with only a subset of columns may be too large to solve. The issue is that the number of constraints may be excessive. To remedy this, if arc costs do not vary by commodity, we can reduce the number of constraints in the model using an alternative formulation called the tree formulation, as presented in Jones[38]. The idea is to aggregate
commodities with the same origin into a single super commodity\textsuperscript{1}. Each super commodity \(s_{O(k)}\) in the set \(S\) of super commodities corresponds to the set of commodities \(k \in K\) originating at \(O(k)\). Since each commodity \(k \in s_{O(k)}\) may flow along several paths between its origin and its destination, each super commodity \(s_{O(k)}\) can flow along several trees, denoted by the set of trees \(Q^{s_{O(k)}}\). We let \(\Gamma_{ij}^q\) equal 1 if tree \(q\) contains arc \((i, j)\) and equal 0 otherwise. Each tree \(q \in Q^{s_{O(k)}}\) is rooted at \(O(k)\) and contains one \(O(k)\) to \(D(k)\) path, denoted \(p_q^k\), for only those \(k \in s_{O(k)}\). The flow on each path in a tree is a constant \(w\), with \(0 \leq w \leq 1\), of \(b^k\) for every commodity \(k\) in the tree.

Before presenting the formulation, we define these additional notations.

**SETS**

\(s_i\) : \(\{k | O(k) = i, k \in K\}\), a super-commodity comprised of all commodities \(k \in K\) originating at \(i\)

\(S(\exists s_i)\) : \(\{s_i | i \in O(k), k \in K\}\), set of super-commodities \(s_i\), \(i \in O(k)\), for all \(k \in K\)

\(q\) : \(\{(i, j) | (i, j) \in p_q^k, p_q^k \in P^k, k \in s_{O(k)}\}\), a tree comprised of paths \(p_q^k\), \(p_q^k \in P^k, k \in s_{O(k)}\)

\(p_q^k\) : a path from \(O(k)\) to \(D(k)\) in tree \(q\)

\(Q^{s_{O(k)}}(\exists q)\) : set of trees originating at \(O(k)\), for all super commodities \(s_{O(k)} \in S\)

**PARAMETERS**

\(c_q^{s_{O(k)}}\) : total cost of sending \(b^k\) units of \(k \in s_{O(k)}\) along the path \(p_q^k\), for all \(q \in Q^{s_{O(k)}}\) and all \(s_{O(k)} \in S\)

**DECISION VARIABLES**

\(w_q^{s_{O(k)}}\) : fraction of \(b^k\) assigned to path \(p_q^k\) for each \(k \in s_{O(k)}\), for all \(q \in Q^{s_{O(k)}}\) and all \(s_{O(k)} \in S\)

The tree formulation for the MCF problem is:

\[
\min \sum_{s_{O(k)} \in S} \sum_{q \in Q^{s_{O(k)}}} c_q^{s_{O(k)}} w_q^{s_{O(k)}} \quad (5.9)
\]

\[
\sum_{q \in Q^{s_{O(k)}}} w_q^{s_{O(k)}} = 1 \quad \text{for all } s_{O(k)} \in S \quad (5.10)
\]

\textsuperscript{1}Alternatively, super commodities can be aggregated by destination.
\[
\sum_{s_{O(k)} \in S} \sum_{q \in Q^{s_{O(k)}}} \left( \sum_{k \in s_{O(k)}} \Gamma_{ij} b^k \right) w_{q}^{s_{O(k)}} \leq u_{ij} \quad \text{for all } (i, j) \in A \quad (5.11)
\]

\[
w_{q}^{s_{O(k)}} \geq 0 \quad \text{for all } q \in Q^{s_{O(k)}}, \text{ for all } s_{O(k)} \in S \quad (5.12)
\]

As before, the objective is to minimize the total cost of flowing commodities from their origins to their destinations. Constraints (5.10) ensure that exactly the total quantity of each commodity is assigned to the network. Constraints (5.11) ensure that the total arc flow does not exceed arc capacity, for each \((i, j) \in A\) and constraints (5.12) ensure that tree flows are all nonnegative.

Just as in the comparison of node-arc and path formulations, the number of constraints in the tree formulation is reduced compared to the number in the path formulation \((|S| + |A|\), with \(|S|\) equal to the number of super commodities, or equivalently, the number of origins vs. \(|K| + |A|\)), and the number of variables in the tree formulation is increased exponentially relative to the number in the path formulation (the number of variables in the tree formulation is the number of ways to combine path variables for each commodity.)

### 5.1.6 Tree Solution

As with the path formulation, column generation can be used to solve the tree formulation. The pricing problem, however, has the following modified form:

\[
z^* = \operatorname{Minimize} \sum_{q \in Q^{s_{O(k)}}, s_{O(k)} \in S} c_{q}^{s_{O(k)}} - \sigma_{s_{O(k)}} - \sum_{(i, j) \in A} \left( \sum_{k \in s_{O(k)}} \Gamma_{ij} b^k \right) \pi_{ij}
\]

where \(\pi\) are non-positive and \((\sigma, \pi)\) are the dual variables associated with constraints (5.10) and (5.11), respectively. Again, for the reasons stated above, if \(z^* \geq 0\) then the tree formulation LP is solved. However, if \(z^* < 0\), the solution to the pricing problem identifies a tree that may improve the current solution if added to the restricted LP.

Rewriting the pricing problem:
\[
\begin{align*}
    z^* &= \operatorname{Minimize}_{q \in Q^{s_{O(k)}}, s_{O(k)} \in S} \sum_{k \in s_{O(k)}} c_{k}^{b_{k}} - \sum_{(i,j) \in A} \left( \sum_{k \in s_{O(k)}} p_{ij}^{k} b_{k} \right) \pi_{ij} - \sigma_{s_{O(k)}} \\
    &= \operatorname{Minimize}_{q \in Q^{s_{O(k)}}, s_{O(k)} \in S} \sum_{k \in s_{O(k)}} c_{k}^{b_{k}} - \sum_{(i,j) \in A} \left( \sum_{k \in s_{O(k)}} q_{ij}^{k} b_{k} \right) \pi_{ij} - \sigma_{s_{O(k)}} \\
    &= \operatorname{Minimize}_{q \in Q^{s_{O(k)}}, s_{O(k)} \in S} \sum_{(i,j) \in A} \sum_{k \in s_{O(k)}} c_{ij}^{k} b_{k} \delta_{ij}^{k} - \sum_{(i,j) \in A} \left( \sum_{k \in s_{O(k)}} q_{ij}^{k} b_{k} \right) \pi_{ij} - \sigma_{s_{O(k)}} \\
    &= \operatorname{Minimize}_{q \in Q^{s_{O(k)}}, s_{O(k)} \in S} \sum_{(i,j) \in A} \sum_{k \in s_{O(k)}} \left( c_{ij}^{k} - \pi_{ij} \right) \delta_{ij}^{k} b_{k} - \sigma_{s_{O(k)}}
\end{align*}
\]

Again, if for every \( s_{O(k)} \in S \), \( \operatorname{Minimize}_{q \in Q^{s_{O(k)}}} \sum_{(i,j) \in A} \sum_{k \in s_{O(k)}} \left( c_{ij}^{k} - \pi_{ij} \right) \delta_{ij}^{k} b_{k} \geq \sigma_{s_{O(k)}} \), then optimality is achieved. This problem is a shortest path problem for each super commodity \( s_{O(k)} \in S \) over the service network with modified link costs of \( \left( c_{ij}^{k} - \pi_{ij} \right) \), for all \((i,j) \in A\).

Solving the pricing problem for the tree formulation, like the path formulation, requires only the solution of one shortest path problem for each super commodity \( s_{O(k)} \in S \), since a shortest path algorithm actually produces a shortest path tree from its origin to all destinations. So, even with the change in variables from paths to trees, the pricing problem remains the same.

Jones, et al.\cite{38} show that given costs that are not commodity specific, the optimal solution values for the path and tree formulations are equal and tree solutions are feasible for the path and node-arc formulations. To show the equivalence of the two models, we provide the following two theorems.

**Theorem 8** Each basic solution to the tree formulation corresponds to a feasible solution to the path formulation, and their objective function values are equal.

**Proof.** Let \( w_{q}^{s_{O(k)}} \), for all \( q \in Q^{s_{O(k)}} \) and \( s_{O(k)} \in S \), denote a basic feasible solution to the tree formulation. Then for each \( p \in P^{k} \) and \( k \in O(k) \), we construct a feasible solution, denoted \( x_{p}^{k} \), as follows:

\[
    x_{p}^{k} = \sum_{q \in Q^{s_{O(k)}}} w_{q}^{s_{O(k)}} \lambda_{p}^{q,k}, \forall p \in P^{k}, k \in K, \text{where } \lambda_{p}^{q,k} = 1 \text{ if } p \in P^{k} \text{ is contained in tree } q \text{ and 0 otherwise, for } \forall q \in Q^{s_{O(k)}}, s_{O(k)} \in S.
\]

**Theorem 9** Each basic solution to the path formulation corresponds to a feasible solution to the tree formulation, and their objective function values are equal.
\[
\begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|c|}
\hline
\text{iteration} & \alpha & q^{iter} & \rho^1_1 & \rho^1_2 & \rho^1_3 & \rho^2_1 & \rho^2_2 & \rho^2_3 & \text{Path Solution} \\
\hline
\text{iter} = 0 & - & - & x_{11} & x_{22} & x_{11} & x_{22} & x_{33} & \text{0.4} & \text{0.6} & \text{0.3} & \text{0.2} & \text{0.5} \\
\text{iter} = 1 & 0.2 & q^1 & 0.2 & - & - & 0.2 & - & \text{0.2} & \text{0.6} & \text{0.3} & - & \text{0.5} \\
\text{iter} = 2 & 0.2 & q^2 & 0.2 & - & 0.2 & - & - & - & \text{0.6} & \text{0.1} & - & \text{0.5} \\
\text{iter} = 3 & 0.1 & q^3 & - & 0.1 & 0.1 & - & - & - & \text{0.5} & - & - & \text{0.5} \\
\text{iter} = 4 & 0.5 & q^4 & - & 0.5 & - & - & 0.5 & - & - & - & - & - \\
\hline
\end{array}
\]

Table 5.1: Deriving a Tree Solution from a Feasible Path Solution

**Proof.** Let \( x_p^k, \forall p \in P^k, k \in K \) represent a basic solution to the path formulation. Then, for each origin \( i \), we construct a set of trees originating at \( i \), denoted \( Q^*_i \), with \( Q^*_i = 0 \) initially:

1) Let \( \rho_k = \left\{ p' \mid x_{p'}^k \leq x_p^k, p \in P^k \right\}, \forall k \in s_i \) and \( \alpha = \min \{ \rho_k \mid k \in s_i \} \).

2) Let \( q = \cup_{k \in s_i} \rho_k \) with \( w_q^i = \alpha \).

3) Let \( Q^*_i = Q^*_i \cup q \).

4) Set \( x_{p_k}^k = x_{p_k}^k - \alpha, \forall k \in s_i \).

We repeat steps 1-4 until \( \alpha = 0 \). Since \( \sum_{p \in P^k} x_p^k = 1, \forall k \in K \), then \( x_p^k = 0, \forall p \in P \), \( k \in s_i \) when \( \alpha = 0 \). So \( w_q^i \), for all \( q \in Q^*_i \) and \( i \in O(k), k \in K \), represents a feasible solution to the tree formulation. \( \blacksquare \)

**Example** Consider the random network in Fig.5-1, consisting of 4 nodes and 5 arcs \((a_1, a_2, a_3, a_4, a_5)\) with labels \((\alpha, \beta)\) denoting arc cost of commodity and arc capacity, respectively. 10 units of commodity 1 originate at node 1 and are destined to node 3 and 15 units of commodity 2 originate at node 1 and are destined to node 4. Suppose that we have a feasible path solution containing 2 paths for commodity 1 and 3 paths for commodity 2. Let's assume that \( p_1^1 = \{a_1, a_3\}, p_1^2 = \{a_2\}, p_1^3 = \{a_1, a_4\}, p_2^1 = \{a_1, a_3, a_5\}, p_2^2 = \{a_2, a_5\} \), with corresponding solution values are \( x_1^1 = 0.4, x_1^2 = 0.6, x_1^3 = 0.3, x_2^1 = 0.2 \) and \( x_2^2 = 0.5 \), respectively.

Then, as detailed in Table 5.1, the following results are achieved by applying the algorithm detailed in the proof for Theorem 2.
Figure 5-1: Random Network Example

- Iteration 1: $\alpha = 0.2$, $q = \{p_1^1, p_2^1\}$, $w_1^1 = 0.2$
- Iteration 2: $\alpha = 0.2$, $q = \{p_1^1, p_1^2\}$, $w_2^1 = 0.2$
- Iteration 3: $\alpha = 0.1$, $q = \{p_2^3, p_1^2\}$, $w_3^1 = 0.1$
- Iteration 4: $\alpha = 0.5$, $q = \{p_2^3, p_3^3\}$, $w_4^1 = 0.5$

Table 5.1 shows details of the algorithm.

The difference is the rate at which the column generation algorithm converges to the optimal solution. Jones et al. [38] report much slower convergence for the tree formulation than for the path formulation. That is, substantially fewer master problem iterations are needed to solve the path formulation than the tree formulation. To illustrate, reconsider the random network in Fig.5-1.

In Table 5.2 we contrast and summarize the results of the column generation procedure when applied to the path and tree formulations. $d, x, w$ denote artificial variables, path variables and tree variables, respectively. The Case 1 example shows that an additional iteration is required to solve the tree formulation compared to the path formulation. The difference results because the tree formulation is inherently less flexible than the path formulation. To illustrate, notice
that the path formulation allowed commodity 1 to be assigned in its entirety to \( \{a2\} \) but the tree variable allowed only \( 3/5 \) of commodity 1 to be assigned to that same path. The issue is that the tree formulation couples the flow assignment decisions for all commodities: the full assignment of commodity 1 to \( \{a2\} \) cannot occur because commodity 2 cannot be fully assigned to \( \{a1,a2\} \) due to the limiting capacity of \( a4 \); and \( w2 \) forces commodities 1 and 2 to share the capacity of arc \( a2 \).

In Case 2 and Case 3, we show that convergence of the tree formulation is largely dependent on the network characteristics. When the network is less congested, or equivalently, network capacity is under utilized, the gap between the number of master problem iterations for the path formulation and for the tree formulation may be reduced substantially. For example, when we increase the arc capacity of either \( a2 \) from 10 to 20 (Case 2 in Table 5.2), or \( a4 \) from 5 to 15 (Case 3 in Table 5.2), there is no difference between the path and tree formulations in terms of the number of master problem iterations or the number of paths generated.

### 5.2 Computational Results
Using data from a large express shipment carrier, we compared the solution of the node-arc, path and tree formulations, using an IBM RS/6000, 370 workstation with 256 MB RAM. The carrier's service network contained 807 nodes, 1,363 links and 17,539 origin/destination specific commodities (see Table 5.3). The number of super commodities, defined by the number of origins, is 136. Only 292 out of 1363 links are capacitated because ground movement capacity is effectively unlimited, since it is relatively cheap, and only aircraft capacity is limited.

<table>
<thead>
<tr>
<th></th>
<th>807</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nodes</td>
<td></td>
</tr>
<tr>
<td>Links</td>
<td>1,363</td>
</tr>
<tr>
<td>(capacitated)</td>
<td>292</td>
</tr>
<tr>
<td>(uncapacitated)</td>
<td>1,071</td>
</tr>
<tr>
<td>No. of Commodities</td>
<td>17,539</td>
</tr>
<tr>
<td>(No. of super commodities)</td>
<td>136</td>
</tr>
</tbody>
</table>

Table 5.3: MCF Data Set Size

For this problem, the resulting sizes of the three MCF formulations are shown in Table 5.4. As a minor modeling enhancement, we deleted all uncapacitated links from the bundle constraints since their inclusion is not necessary. This reduces the number of constraints for the path and tree formulations from 18,902 to 17,832 and from 1,499 to 428, respectively. It is important to keep in mind that the number of constraints in the problem plays a critical role in determining the speed of the LP solver.

<table>
<thead>
<tr>
<th></th>
<th>no. of constraints</th>
<th>no. of variables</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>all links</td>
<td>capacitated links only</td>
</tr>
<tr>
<td>Node-Arc</td>
<td>14,155,336</td>
<td>14,154,265</td>
</tr>
<tr>
<td>Path</td>
<td>18,902</td>
<td>17,832</td>
</tr>
<tr>
<td>Tree</td>
<td>1,499</td>
<td>428</td>
</tr>
</tbody>
</table>

Table 5.4: MCF Matrix Size

We were unable to solve the node-arc formulation due to limitations in the size of our random access memory, however, we solved the path formulation in about 40 minutes and, contrary to the experiences reported by Jones, et al.[38], we solved the tree based model within
1 minute. We believe that the tree model was solved so quickly in this application because the network is relatively uncongested. Since the express package service problem has tight service requirements, maximum consolidation and utilization of capacity is not possible. Instead, planes are forced to fly in order to satisfy service commitments. Lack of congestion together with a relatively small number of constraints (428 vs. 17,832 in the path formulation), allows the column generation algorithm to terminate quickly without tailing.
Chapter 6

Service Network Design for Express Package Delivery

In this chapter, we present alternative models and algorithms for the service network design problem of a large express package delivery operation. We first present our exact models for the express package service network design problem, called EPSND_Path and EPSND_Tree, and then describe our solution algorithm. Next, we present an enhanced model, called EPSND-Cut, and its solution algorithm. Finally, we describe an approximate model, called AEPSND-Cut, and solution approach. We evaluate these various models and algorithms using data provided by the express package delivery company.

6.1 Express Package Delivery: Baseline Model and Solution Approach

In the context of multimodal express package service network design (EPSND), the design variables in the service network design models presented in Chapter 2 represent jet and feeder movements and the flow variables describe package flows. In each case, these variables correspond to routes, and not single legs. We adopt this route-based variable definition for three major reasons:
1. We easily can capture fixed costs and other nonlinear costs since each aircraft/vehicle is assigned to at most one design variable.

2. Complex restrictions on aircraft, vehicle or package routings are satisfied by each variable and need not be represented by constraints that are often difficult or even impossible to write.

3. The node-arc express package delivery formulation is prohibitively large (with hundreds of millions of decision variables and more than 42 million constraints) and impossible to solve.

The disadvantage of route-based variables, however, is that the number of variables explodes and so, specialized solution procedures, like column generation must be used.

Given our variable definitions, the objective of our EPSND problem is to find:

1. Jet and feeder routes that minimize fixed design and variable operating costs; and

2. Package flow routes that satisfy customer demands without violating service restrictions and capacity limits imposed on each design leg.

6.1.1 Side Constraints

The SNDP formulation must be augmented to reflect particular express package delivery operational constraints, including fleet balance, fleet size, hub sort capacity, hub landing capacity, and connectivity.

Fleet Balance

*Fleet balance constraints* force the number of routes into a location for a particular fleet to be equal to the number out, for each fleet $f \in F$:

$$
\sum_{r \in R^f \cup R^d} \beta^r_i y^f_r = 0 \quad \text{for all } i \in N, \ f \in F
$$

(6.1)

where $\beta^r_i$ is equal to 1 if route $r$ ends at node $i$, is equal to $-1$ if route $r$ begins at $i$, and is equal to 0 otherwise.
Fleet Size

Each fleet has a limited number of aircraft/vehicles and so, the number of aircraft/vehicles of a particular type used may not exceed the number available. We model these fleet size constraints by restricting the number of pickup routes or the number of delivery routes for fleet \( f \) to be less than \( n^f \), the fleet size, in \( f \in F \):

\[
\sum_{r \in R^f_p} y^f_r \leq n^f \quad f \in F; \text{ or}
\sum_{r \in R^f_d} y^f_r \leq n^f \quad f \in F.
\]

We do not need to include both sets of constraints since fleet balance is ensured by (6.1).

Hub Sort Capacity

Each hub is able to sort a limited number of packages per time period. To model these hub sort capacity constraints, for each hub, we divide the total time during which sorting is performed into equal-length intervals \( t = \{1, 2, ..., T\} \). We let \( P^f_i \) be the set of package routes with earliest arrival time at hub \( i \) later than or equal to the start time of interval \( t \in \{1, 2, ..., T\} \) and \( e^m_i \) be the sort capacity of hub \( i \) during interval \( t \in T \). Then, for the path formulation, the hub sort capacity constraints are:

\[
\sum_{k \in K} \sum_{p \in P^k \cap P^t_i} b^k_x^{kp} \leq \sum_{m=t}^T e^m_i \quad i \in H, \; t \in \{1, 2, ..., T\}.
\]

We can represent the same constraints for the tree formulation as:

\[
\sum_{s \in S} \sum_{q \in Q^s} \left( \sum_{k \in K} \sum_{p \in P^k \cap P^t_i} \left( \delta^p_{ij} b^k \right) \right) w^s_q \leq \sum_{m=t}^T e^m_i \quad i \in H, \; t \in \{1, 2, ..., T\}.
\]

To illustrate, assume that there are five packages arriving at a hub with 3 hours of sort time and a sort capacity of 2 packages per hour. Each package is represented by an arc in Figure (6-1), with its head pointing to its earliest arrival time at the hub.
The corresponding hub sort capacity constraints are:

- **time_interval_1**: \( x_1 + x_2 + x_3 + x_4 + x_5 \leq 6 \)
- **time_interval_2**: \( x_3 + x_4 + x_5 \leq 4 \)
- **time_interval_3**: \( x_4 + x_5 \leq 2 \)

**Hub Landing Capacity**

Each hub has a landing capacity that limits the number of aircraft that can land in an interval of time. To model these hub landing capacity constraints, for each hub, we divide the total time during which aircraft are arriving into equal-length intervals \( t = \{1, 2, \ldots, T\} \). We let \( L_i^t \) denote the set of pickup routes with earliest arrival time at hub \( i \) no earlier than the start time of interval \( t \), and \( a_i^t \) be the maximum number of aircraft that can land (i.e., the landing capacity) at hub \( i \) during interval \( t \), for all \( t \in T \). Then, for either the path or tree formulations, the hub landing capacity constraints can be represented as:

\[
\sum_{f \in F} \sum_{r \in R_f \cap L_i^t} y_{fr}^t \leq \sum_{m=t}^{T} a_i^m \quad i \in H, \ t \in \{1, 2, \ldots, T\}.
\]

**Connectivity**

Many package delivery networks contain a major hub (for example, Memphis, Tennessee for Federal Express, Louisville, Kentucky for United Parcel Service, and Wilmington, Ohio for Airborne Express) and one or more regional hubs. For operational reasons, some carriers require all-point service to and from their major hub. This means that for every origin location, there should be at least one route from that location to the major hub and similarly, for every
destination location, there should be at least one route to it from the major hub. The idea is that with all the aircraft/vehicles arriving and departing the major hub, achieving timely service should still be possible, even with service disruptions.

We let $V_P$ represent the set of pickup routes ending at the major hub and let $V_D$ represent the set of delivery routes beginning at the major hub. We model these all-point service connectivity constraints for both the path and tree formulations as:

$$\sum_{f \in F} \sum_{r \in R'_p \cap V_P} y^f_r \geq 1 \quad i \in N$$

$$\sum_{f \in F} \sum_{r \in R'_D \cap V_D} y^f_r \geq 1 \quad i \in N.$$

6.1.2 Express Package Service Network Design Models

The resulting path ($EPSND\_Path$) and tree ($EPSND\_Tree$) formulations for the express package service network design problem are:

($EPSND\_Path$)

$$\min \sum_{r \in R'_p \cup R'_D} h^f_r y^f_r + \sum_{k \in K} \sum_{p \in P^k} \left( c_p^k b^k \right) x^k_p$$

$$\sum_{k \in K} \sum_{p \in P^k} \left( \delta^f_{ij} b^k \right) x^k_p \leq \sum_{f \in F} \sum_{r \in R'_p \cup R'_D} u^f y^f_r \alpha^f_{ij} \quad \text{for all } (i, j) \in A \quad (6.3)$$

$$\sum_{r \in R'_p \cup R'_D} \beta^f_r y^f_r = 0 \quad \text{for all } i \in N, \ f \in F \quad (6.4)$$

$$\sum_{r \in R'_p} y^f_r \leq n^f \quad f \in F \quad (6.5)$$

$$\sum_{f \in F} \sum_{r \in R'_p \cap L^t} y^f_r \leq \sum_{m=t}^{T} \alpha^m_i \quad i \in H, \ t \in \{1, 2, ..., T\} \quad (6.6)$$
\[
\sum_{f \in F} \sum_{r \in R_p \cap W_p} y_r^f \geq 1 \quad i \in N \tag{6.7}
\]

\[
\sum_{f \in F} \sum_{r \in R_D \cap W_D} y_r^f \geq 1 \quad i \in N \tag{6.8}
\]

\[
\sum_{p \in P^k} x_p^k = 1 \quad \text{for all } k \in K \tag{6.9}
\]

\[
\sum_{k \in K} \sum_{p \in P^k \cap P_i} b^k x_p^k \leq \sum_{m=1}^{T} e_i^{m} \tag{6.10}
\]

\[
x_p^k \geq 0 \quad \text{for all } k \in K, \ p \in P^k \tag{6.11}
\]

\[
y_r^f \geq 0 \text{ and integer} \quad \text{for all } f \in F, \ r \in R_p' \cup R_D' \tag{6.12}
\]

**EPSND Tree**

\[
\min \quad \sum_{f \in F} \sum_{r \in R_p' \cup R_D'} h_r^f y_r^f + \sum_{s \in S} \sum_{q \in Q^s} \left( \sum_{k \in K} \sum_{p \in P^k} c_p^{k, q} b^k \right) w_q^s \tag{6.13}
\]

\[
\sum_{s \in S} \sum_{q \in Q^s} \left( \sum_{k \in K} \sum_{p \in P^k} \Gamma_{ij}^q b^k \right) w_q^s \leq \sum_{f \in F} \sum_{r \in R_p' \cup R_D'} \alpha_{ij}^q u_r^f y_r^f \quad \text{for all } (i, j) \in A \tag{6.14}
\]

\[
\sum_{r \in R_p' \cup R_D'} \beta_r^f y_r^f = 0 \quad \text{for all } i \in N, \ f \in F \tag{6.15}
\]

\[
\sum_{r \in R_p'} y_r^f \leq n_f \quad \text{for } f \in F \tag{6.16}
\]

\[
\sum_{f \in F} \sum_{r \in R_p \cap L_i} y_r^f \leq \sum_{m=1}^{T} a_i^m \quad i \in H, \ t \in \{1, 2, \ldots, T\} \tag{6.17}
\]
\[ \sum_{f \in F} \sum_{r \in R'_p \cap V_p} y^f_r \geq 1 \quad i \in N \quad (6.18) \]

\[ \sum_{f \in F} \sum_{r \in R'_d \cap V_d} y^f_r \geq 1 \quad i \in N \quad (6.19) \]

\[ \sum_{q \in Q^s} w^s_q = 1 \quad \text{for all } s \in S \quad (6.20) \]

\[ \sum_{s \in S} \sum_{q \in Q^s} \left( \sum_{k \in K} \sum_{p \in P \cap P'_l} \left( \delta^p_{ij} b^k \right) \right) w^s_q \leq \sum_{m=1}^{T} e^m_i \quad i \in H, \ t \in \{1, 2, ..., T\} \quad (6.21) \]

\[ w^s_q \geq 0 \quad \text{for all } q \in Q^s, \ s \in S \quad (6.22) \]

\[ y^f_r \geq 0 \text{ and integer} \quad \text{for all } f \in F, \ r \in R'_p \cup R'_d \quad (6.23) \]

The objective function (6.2), (6.13) is to find the cost minimizing set of aircraft, vehicle and package routes. Constraints (6.3), (6.14) are forcing constraints that assign packages to a leg only if that leg is part of a selected aircraft/vehicle route and the leg's capacity is not exceeded by the assignment. With respect to the route assignment variables, constraints (6.4) and (6.15) are balance constraints, constraints (6.5) and (6.16) are fleet size constraints, constraints (6.6) and (6.17) are the hub landing capacity constraints, and constraints (6.7), (6.8) and (6.18), (6.19) are the connectivity constraints. With respect to the package flow variables, constraints (6.9) and (6.20) are the demand constraints that ensure that all shipments are serviced and constraints (6.10) and (6.21) are the hub sort capacity constraints. Constraints (6.11) and (6.22) ensure nonnegativity of package flows and (6.12) and (6.23) limit the design variables to nonnegative integer values.

### 6.1.3 Express Package Service Network Design Model Size
### Table 6.1: Data Set Size

<table>
<thead>
<tr>
<th></th>
<th>DS1</th>
<th>DS2</th>
<th>DS3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locations</td>
<td>31</td>
<td>91</td>
<td>141</td>
</tr>
<tr>
<td>Hubs</td>
<td>4</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>O/D (tree)</td>
<td>863 (30)</td>
<td>6,157 (80)</td>
<td>17,651 (136)</td>
</tr>
<tr>
<td>Commodities</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Volume</td>
<td>70,156</td>
<td>404,241</td>
<td>852,362</td>
</tr>
<tr>
<td>Number of</td>
<td>Jets</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Fleet Types</td>
<td>Feeder</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

### Table 6.2: Number of Constraints in DS3

<table>
<thead>
<tr>
<th></th>
<th>EPSND_Path</th>
<th>EPSND_Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>forcing constraints</td>
<td>151,002</td>
<td>151,002</td>
</tr>
<tr>
<td>fleet balance</td>
<td>705</td>
<td>705</td>
</tr>
<tr>
<td>fleet size</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>hub landing capacity</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td>connectivity</td>
<td>282</td>
<td>282</td>
</tr>
<tr>
<td>shipment demand</td>
<td>17,651</td>
<td>136</td>
</tr>
<tr>
<td>hub sort capacity</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td>total</td>
<td>169,699</td>
<td>152,184</td>
</tr>
</tbody>
</table>

The express package delivery company provided three different data sets representing their express package delivery operation (Table 6.1). DS3 represents their entire operation, with DS1 and DS2 representing portions of their operation. We use DS1 and DS2 primarily for computational testing, and for gaining insight about the larger DS3 problem.

A time-space network for DS3 is huge and impractical to use. We use instead our new derived schedule network representation in building our route-based model and solution approach. Even using the derived schedule approach, the EPSND_Path and EPSND_Tree models for DS3 are still very large (Table 6.2) with over one hundred thousand constraints and over one billion decision variables (i.e., 1.0 billion package flow variables and 0.3 million fleet design set variables). This size LP is too large for state-of-the-art LP/IP solvers (such as, CPLEX[20], OSL[36], MINTO[56], etc.). After applying the network reduction methods of node and link consolidation, we are able to reduce the number of constraints of EPSND_Tree from 152,184 to 16,537 (Table 6.3) while maintaining exactness of the formulation.


<table>
<thead>
<tr>
<th></th>
<th>DS1 before</th>
<th>DS1 after</th>
<th>DS2 before</th>
<th>DS2 after</th>
<th>DS3 before</th>
<th>DS3 after</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Consolidation</td>
<td># nodes</td>
<td>5,014</td>
<td>190</td>
<td>19,368</td>
<td>847</td>
<td>75,944</td>
</tr>
<tr>
<td></td>
<td># links</td>
<td>9,826</td>
<td>4,053</td>
<td>38,162</td>
<td>19,396</td>
<td>151,002</td>
</tr>
<tr>
<td>Link Consolidation</td>
<td># links</td>
<td>4,053</td>
<td>1,242</td>
<td>19,396</td>
<td>5,029</td>
<td>72,144</td>
</tr>
</tbody>
</table>

Table 6.3: Node and Link Consolidation Results

6.1.4 Express Package Service Network Design Solution

We solve EPSND_TREE using branch-and-bound with the root node LP solved using column generation. This is a heuristic approach since we generate columns only at the root node, and to obtain an optimal solution it may be necessary to generate additional columns at other nodes in the branch-and-bound tree.

The solution algorithm for the EPSND_TREE LP relaxation is depicted in Figure 6-2. We use explicit column generation for both pickup and delivery design variables and we use implicit column generation for package flow variables. Our solution procedure begins with a very small restricted master problem (RMP) containing only dummy variables. We explicitly price out design variables and add them to RMP if necessary. Next, we implicitly price out package flow variables and add them to RMP if necessary. We repeat this process until no negative reduced cost columns are found.

For the design variables, we compute the reduced cost of each route for each fleet and add some number of variables with negative reduced cost. Explicit pricing is quite efficient since there are relatively few design variables (about 0.3 million). Implicit pricing for these design variables, however, is very inefficient because the pricing subproblem cannot be modeled as an easy problem. The difficulty is that dual prices correspond to routes and not arcs.

Only implicit column generation is practical for the package flow variables since there are about 1.0 billion of them. The hub sort capacity constraints make implicit pricing difficult, however. Without them, the pricing subproblem is for each commodity $k$, a shortest path problem over the derived schedule network with modified link costs of $\left(c^{k}_{ij} - \pi_{ij}\right)$, for all $(i, j) \in A$, where $\pi$ is the vector of dual variables associated with the forcing constraints. When the hub sort capacity constraints are included, their associated dual variables can't be assigned to arcs in the derived schedule network since they correspond to routes in the network. The
Figure 6-2: Root Node LP Solution Procedure
pricing subproblem, then, is greatly complicated and can no longer be solved using shortest path algorithms. The result is likely to be a tremendous increase in solution time since the pricing subproblem is solved so many times. Consequently, we ignore the sort capacity constraints and try to capture their effects by enforcing the landing capacity constraints at hubs.

6.1.5 Express Package Service Network Design Computational Results

We are able to solve the LP relaxation of EPSND_Tree within 40 minutes on a SGI Power Challenge 2-processor workstation with 256 MB RAM using CPLEX version 4.0[20]. The LP bound generated, however, is very loose and ineffective. In fact, the best IP solution obtained after more than two days of CPU time using branch-and-bound was more than three times the value of the LP solution.

6.2 Express Package Delivery: Enhanced Model and Solution Approach

Our experience in solving EPSND_Tree is consistent with the experiences of others who conclude that linear programming relaxations of network design problems provide very poor lower bounds on the optimal integer solution values (Magnanti[47], [49], [51] and Padberg[59]). The poor LP bound limits our ability to solve EPSND_Tree, so, like others, we strengthen our LP relaxation by adding cuts.

6.2.1 Cutset Inequalities

To strengthen EPSND_Tree, we add aggregate capacity demand inequalities (as detailed in Chapter 2). For this application, these inequalities require that the total capacity provided by the design variables be larger than or equal to the total demand, for any O-D cutset \{S, T\}. We define an O/D cutset \{S, T\} by a partitioning of the node set \(N\) into two nonempty disjoint sets \(S \subset N\) and \(T = N\setminus S\). An arc \((i, j)\) belongs to cutset \{S, T\} if nodes \(i\) and \(j\) belong to different sets \(S\) and \(T\). Aggregate demand, \(D_{S,T}\), denotes the demand of all commodities with origin and destination in different subsets, i.e., with \(O(k) \in S\) and \(D(k) \in T\) or \(O(k) \in T\) and \(D(k) \in S\). Letting \(Y^{S,T}_i = \sum_{r \in R} \sum_{(i,j) \in \{S,T\} \cap r} y^r_{ij}\) and \(D_{S,T}\) denote the total demand from
the set $S$ to the set $T$, the aggregate capacity demand inequalities are:

$$\sum_{f \in F} w_f y_{ST} \geq D_{ST} \quad \text{for all } O/D \text{ cutsets } \{S,T\}. \quad (6.24)$$

We strengthen (6.24) by lifting to create either Chvátal-Gomory (C-G) cuts or cutset inequalities (see Chapter 2). Generally, cutset inequality lifting provides stronger LP bounds to EPSND when there are multiple fleet types available; however, due to the computational difficulty of the cutset inequality lifting procedure, we lift aggregate capacity demand inequalities (6.24) with C-G cuts when the number of fleet types exceeds two. Otherwise, we lift them with cutset inequalities. For ease of exposition, we let inequality (6.24) represent both the original aggregate capacity demand inequalities plus lifted inequalities and we refer to them as cutset inequalities.

6.2.2 EPSND-Cut Model

Our enhanced model for the EPSND problem is:

**EPSND-Cut**

$$\min \sum_{f \in F} \sum_{r \in R_p \cup R_D} h_{r'}, y_{r'} + \sum_{s \in S} \sum_{q \in Q^s} \left( \sum_{k \in E} \sum_{p \in F^k} c_{p}^{k} b_{p}^{k} \right) w_{q}^{s} \quad (6.25)$$

$$\sum_{s \in S} \sum_{q \in Q^s} \left( \sum_{k \in E} \sum_{p \in F^k} \Gamma_{ij}^{q} b_{ij}^{k} \right) \frac{w_{q}^{s}}{\sum_{f \in F} \sum_{r \in R_{p} \cup R_{D}} \alpha_{ij}^{q} u_{r}^{q} y_{r}^{q}} \leq \sum_{f \in F} \sum_{r \in R_{p} \cup R_{D}} \alpha_{ij}^{q} u_{r}^{q} y_{r}^{q} \quad \text{for all } (i,j) \in A \quad (6.26)$$

$$\sum_{r \in R_{p} \cup R_{D}} \beta_{r}^{q} y_{r}^{q} = 0 \quad \text{for all } i \in N, \quad f \in F \quad (6.27)$$

$$\sum_{r \in R_{p}} y_{r}^{f} \leq n^{f} \quad f \in F \quad (6.28)$$

$$\sum_{f \in F} \sum_{r \in R_{p} \cap L_{i}^{m}} y_{r}^{f} \leq \sum_{m=t}^{T} a_{i}^{m} \quad i \in H, \quad t \in \{1,2,\ldots,T\} \quad (6.29)$$
\begin{equation}
\sum_{f \in F} \sum_{r \in R^f_p \cap V_p} y^f_r \geq 1 \quad i \in N
\end{equation}

(6.30)

\begin{equation}
\sum_{f \in F} \sum_{r \in R^f_D \cap V_D} y^f_r \geq 1 \quad i \in N
\end{equation}

(6.31)

\begin{equation}
\sum_{q \in Q^s} w^s_q = 1 \quad \text{for all } s \in S
\end{equation}

(6.32)

\begin{equation}
\sum_{s \in S} \sum_{q \in Q^s} \left\{ \sum_{k \in K^s} \sum_{p \in P^s \cap P^t_1} \left( d_{ij}^p \cdot b^k \right) \right\} w^s_q \leq \sum_{m=t}^{T} e^m_i \quad i \in H, \ t \in \{1,2,\ldots,T\}
\end{equation}

(6.33)

\begin{equation}
\sum_{f \in F} u_f Y^S_T \geq D^S_T \quad \text{for all } O/D \text{ cutsets } \{S,T\}
\end{equation}

(6.34)

\begin{equation}
w^s_q \geq 0 \quad \text{for all } q \in Q^s, \ s \in S
\end{equation}

(6.35)

\begin{equation}
y^f_r \geq 0 \text{ and integer} \quad \text{for all } f \in F, \ r \in R^f_p \cup R^f_D
\end{equation}

(6.36)

\textbf{6.2.3 EPSND-Cut Model Size}

The number of constraints in EPSND-Cut corresponds to the number of constraints in EPSND Tree plus the number of cutset inequalities. The number of aggregate capacity demand inequalities themselves is \(2 \left(2^N - 1\right)\). When we consider lifting all of them for each fleet type, then the total number of cutset inequalities becomes \(\left\{2 \left(2^N - 1\right)\right\} \times (|F| + 1)\), where \(|F|\) is the number of fleets and \(N\) denotes the number of locations in the network. Table 6.4 shows the total number of cutset inequalities for the case of two fleet types and a varying number of locations. The first column shows the total number of cuts possible for varying \(N\) and the second column reports the number of cuts possible for varying \(N\) when \(|S| \leq 3\) and \(|T| \leq 3\). For our application with 136 demand locations and 5 jet types, there are \(1.0453 \times 10^{42}\) possible cutset inequalities. Even restricting ourselves to \(|S| \leq 3\) and \(|T| \leq 3\) cuts, there are more than 5 million inequalities.
<table>
<thead>
<tr>
<th>Total No. of Cuts</th>
<th>CUTSET 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N = 10$</td>
<td>$6,138$</td>
</tr>
<tr>
<td>$N = 30$</td>
<td>$6.44 \times 10^9$</td>
</tr>
<tr>
<td>$N = 50$</td>
<td>$6.76 \times 10^{15}$</td>
</tr>
<tr>
<td>$N = 70$</td>
<td>$7.08 \times 10^{21}$</td>
</tr>
<tr>
<td>$N = 100$</td>
<td>$7.61 \times 10^{30}$</td>
</tr>
</tbody>
</table>

Table 6.4: Cutset Inequality Size

We denote $\text{CUT}_r$ as cutset inequalities when $|S| = r$ and $|T| = r$ and $\text{CUTSET}_L$ as $\bigcup_{r=1}^L \text{CUT}_r$.

6.2.4 EPSND-Cut Solution

$\text{EPSND-Cut}$, with its huge number of constraints and its huge number of variables is impossible to solve directly. Instead, we use both row and column generation to solve this problem. Our solution procedure of the $\text{EPSND-Cut}$ LP relaxation, depicted in Figure 6-3, begins with a very small restricted master problem ($RMP$) containing only cutset inequalities with $|S| = 1$ and $|T| = 1$. We explicitly price out design variables and add them to $RMP$ if necessary. Next, we implicitly price out package flow variables and add them to $RMP$ if necessary. Then, we generate additional cutset inequalities by explicitly checking whether there are any violated constraints. (An efficient algorithm for solving the separation problem has not been found, Balakrishnan, et al.[5]). Violated constraints found are added to the $RMP$ and the $RMP$ is solved again. We repeat this process until no violated constraints and no negative reduced cost columns are found. We reduce memory requirements for our $\text{EPSND-Cut}$ algorithm by considering only cutsets with $|S| \leq 3$ and $|T| \leq 3$ (i.e., $\text{CUTSET}_3$).

Implementing row and column generation procedures is a nontrivial task. As we detailed in Chapter 2, one challenge is in adding constraints that do not change the solution procedure for the pricing subproblems and adding variables that do not change the solution procedure for generating violated inequalities. In the latter case, this is a non-issue since we generate cuts with explicit testing of each inequality. For the case of generating columns, we again have no problem with the design variables since they are generated explicitly. The flow variables, however, are generated implicitly and therefore, we must be able to apply the dual variables
corresponding to cuts with nonzero elements in package flow columns to the arcs in the derived schedule network. Otherwise, our shortest path algorithm will not solve the pricing problem. Fortunately, since our cuts involve only design variables, the package flow variables have only zero coefficients in the cutset inequality rows and so, the solution procedure for the package flow pricing subproblem is unaffected by the addition of cuts.

After obtaining root node LP optimal solutions, we use *branch-and-bound* in order to obtain integer solutions. This is a heuristic solution approach since an optimal solution may include columns and/or cuts that are not generated in solving the root node LP.

### 6.2.5 EPSND-Cut Computational Results

With a SGI Power Challenge 2-processor workstation containing 256 MB RAM using CPLEX version 4.0[20], we are unable to achieve an optimal solution to the *EPSND-Cut* LP. The issue is computer memory: we run out of memory before we can add enough cuts to improve significantly the LP relaxation. Our solution then, is to pursue a heuristic approach that utilizes
6.3 **Express Package Delivery: Approximate Model and Heuristic Solution Approach**

Since \textit{EPSND-Cut} is too large to solve, we adopt a heuristic solution approach for the \textit{EPSND} problem. Our approach is to first select a subset of variables for consideration, and second, solve a restricted \textit{EPSND-Cut} model, denoted \textit{REPSND-Cut}, including only the selected variables. The \textit{REPSND-Cut} model is solved using the algorithm for \textit{EPSND-Cut}. Figure 6-4 depicts our heuristic solution approach.

Clearly, the manner in which we select the variables to be included in \textit{REPSND-Cut} has a dramatic effect on the quality of the solution generated. Our selection process involves solving an approximate \textit{EPSND-Cut} model, one in which capacity in the system is modeled inexactly. Our idea is that the solution to this approximate model will provide an indication of which variables should be included in our selected subset. In fact, legs used in the solution to our approximate \textit{EPSND-Cut} model are considered in constructing the set of potential design variables for our restricted \textit{EPSND-Cut} model.

We construct our approximate \textit{EPSND-Cut} model so that its optimal solution will fall into one of two categories, either the solution will be optimal for the \textit{EPSND} problem; or it will be
infeasible because of insufficient capacity to service one or more packages.

Even when the result is infeasibility, we believe that much of the approximate $EPSND-Cut$ solution is likely to be contained in an optimal solution for the $EPSND$ problem.

### 6.3.1 Approximate EPSND-Cut Model

Our approximate $EPSND-Cut$ model, denoted $AEPSND-Cut$, is $EPSND-Cut$ with the package flow variables and associated constraints eliminated.

**(AEPSND-Cut)**

\[
\min \sum_{f \in F} \sum_{r \in R_p \cup R_D} h_r^f y_r^f 
\]

\[
\sum_{f \in F} u_f Y_f^{S,T} \geq D_{S,T} \quad \text{for all } O/D \text{ cutsets } \{S,T\} 
\]

\[
\sum_{r \in R_p \cup R_D} \beta_r^f y_r^f = 0 \quad \text{for all } i \in N, \ , f \in F 
\]

\[
\sum_{r \in R_p} y_r^f \leq n^f \quad f \in F 
\]

\[
\sum_{f \in F} \sum_{r \in R_p \cap L_i} y_r^f \leq \sum_{m=i}^{T} a_i^m 
\text{for } i \in H, \ t \in \{1,2,\ldots,T\} 
\]

\[
\sum_{f \in F} \sum_{r \in R_p \cap V_p} y_r^f \geq 1 
\text{for } i \in N 
\]

\[
\sum_{f \in F} \sum_{r \in R_D \cap V_D} y_r^f \geq 1 
\text{for } i \in N 
\]

\[y_r^f \geq 0 \text{ and integer} \quad \text{for all } f \in F, \ r \in R_p \cup R_D.\]

With only fleet design variables included, the objective function (6.37) is to find the cost-minimizing set of routes satisfying the cutset inequalities (6.38) and constraints (6.39) through

96
(6.44), as defined for *EPSND Tree*.

### 6.3.2 AEPSND-Cut Solution

The algorithm to solve *AEPSND-Cut* is similar to the *EPSND-Cut* algorithm except that we don't have to consider the forcing constraints or generation of the package flow variables. As shown in Figure 6-5, we again avoid memory problems by restricting our generation of cuts to cutset inequalities with $|S| \leq 3$ and $|T| \leq 3$. Using the decision variables generated in the AEPSND-Cut root node LP optimal solution, we obtain an AEPSND-Cut IP solution with the following procedure:

**Step 1:** Solve AEPSND-Cut LP, given a fixed set of constraints, using column generation.

**Step 2:** Achieve a feasible IP solution to AEPSND-Cut, given a fixed set of columns and constraints, using branch-and-bound.

**Step 3:** Explicitly generate violated inequalities, given the current IP primal solution. If there are any violated inequalities found, add them to the current basis and go to Step 1. Otherwise, go to Step 4.

**Step 4:** Generate all route variables that contain at least one link used in the AEPSND-Cut IP solution.

Step 4 of the AEPSND-Cut IP solution procedure generates the only variables considered in the REPSND-Cut model.

### 6.3.3 Bounds on the Heuristic Solution

We let $Z_{*EPSND}^*, Z_{*EPSND-LP}^*$ denote the optimal IP and LP solution values, respectively for *EPSND* and *EPSND LP*, $Z_{*EPSND-Cut}^*, Z_{*EPSND-Cut-LP}^*$ be the optimal IP and LP solution values, respectively for *EPSND-Cut* and *EPSND-Cut LP*, $Z_{*AEPSND-Cut}^*, Z_{*AEPSND-Cut-LP}^*$ be the optimal IP and LP solution values, respectively for *AEPSND-Cut* and *AEPSND-Cut LP*, and $Z_{IP}^H, Z_{LP}^H$ be the best IP and LP solution values respectively for our heuristic approach. Then, the following relationships hold:
Figure 6-5: AEPSND-Cut LP Solution Procedure

- $Z_{AEPSND-Cut}^* \leq Z_{EPSND}^* = Z_{AEPSND-Cut}^* \leq Z_{IP}^H$

- $Z_{AEPSND-Cut-LP}^* \leq Z_{EPSND-Cut-LP}^* \leq Z_{EPSND}^*$

- $Z_{EPSND-LP}^* \leq Z_{EPSND-Cut-LP}^*$

- $Z_{EPSND-LP}^* \geq Z_{AEPSND-Cut-LP}^*$

In all cutset formulations, we add lifted inequalities as well as original aggregate capacity demand inequalities. This is why we are unable to determine the relationship between $EPSND-LP$ and $AEPSND-Cut-LP$.

If the optimal $AEPSND-Cut$ solution is feasible for the $EPSND$ problem, then it solves the $EPSND$ problem, that is $Z_{AEPSND-Cut}^* = Z_{EPSNL-Cut}^* = Z_{EPSND}^*$. If, however, the $AEPSND-Cut$ solution is not feasible, then $Z_{AEPSND-Cut}^*$ provides a lower bound on the optimal solution value, that is, $Z_{AEPSND-Cut}^* \leq Z_{EPSND}^*$.
<table>
<thead>
<tr>
<th></th>
<th>DS1</th>
<th>DS2</th>
<th>DS3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total Cost</strong></td>
<td>Planner’s</td>
<td>267,027</td>
<td>603,039</td>
</tr>
<tr>
<td></td>
<td><em>EPSNDH</em></td>
<td>169,907</td>
<td>567,035</td>
</tr>
<tr>
<td>Savings</td>
<td>36.4%</td>
<td>6.0%</td>
<td>9.54%</td>
</tr>
<tr>
<td><strong>Mileage</strong></td>
<td>Planner’s</td>
<td>28,062</td>
<td>58,622</td>
</tr>
<tr>
<td></td>
<td><em>EPSNDH</em></td>
<td>15,972</td>
<td>44,768</td>
</tr>
<tr>
<td>Savings</td>
<td>43.1%</td>
<td>23.6%</td>
<td>11.85%</td>
</tr>
<tr>
<td><strong># Aircrafts Used</strong></td>
<td>Planner’s</td>
<td>27</td>
<td>71</td>
</tr>
<tr>
<td></td>
<td><em>EPSNDH</em></td>
<td>12</td>
<td>54</td>
</tr>
</tbody>
</table>

Table 6.5: Computational Results

6.3.4 Heuristic Computational Experiences

We implemented our heuristic on a SGI Power Challenge workstation, using CPLEX 4.0[20] with 2 processors and 256 MBs RAM.

To evaluate the performance of our heuristic approach, we compare our solution cost (i.e., $Z_{fp}^H$) for the DS1, DS2, and DS3 problems with the corresponding costs of the solutions generated primarily by hand by planners of the express package delivery company. Our results, summarized in Table 6.5, show that our heuristic approach reduces costs by nearly 10% compared to the planner’s solution for the full-scale operation (DS3). This is a significant cost reduction, translating to tens of millions of dollars annually. In addition to the cost savings, our solution reduces the number of aircraft miles flown by about 12% and uses about 10% fewer aircraft than used in the planner’s solutions. Cost saving comparisons for DS1 and DS2 are not meaningful because the planner’s solution for DS1 (DS2) is just the portion of the planner’s DS3 solution that includes packages in DS1 (DS2). We do not include the savings for the number of aircrafts used because we assume that the aircrafts used are owned by the company. In fact, however, reducing fleet size by 11 aircraft would result in additional savings measuring in the billions of dollars.

The heuristic run times for DS1, DS2 and DS3 are about 20 seconds, 4 hours and 11 hours, respectively. Since these are planning tools, run times of even 11 hours are acceptable, particularly because they result in significant savings.
6.4 Scenario Analysis

In order to gain insight regarding the effects of various problem parameters on the quality of our heuristic solution and on run time, we generate and solve several representative, yet small, test problems that mirror the structure of our EPSND problem. In these test problems, we vary the number of packages, the number of fleets and the number of hubs, and evaluate the effects of these variations.

Our test case network is a subset of DS1 and contains 24 locations, 2 fleet types with the capacity of the second fleet twice that of the first, 3 hubs and 552 commodities (each specified by an origin-destination pair) totaling 70,156 packages. This problem instance is small enough to test all of our solution algorithms and yet, complex enough to include all components of our multimodal express package delivery application.

First, we test the impact of demand variations on the model. Given the original demand volumes, we generate six different demand scenarios, i.e., 50%, 100%, 150%, 200%, 250% and 300%, of the original demand volume. Second, we test the impact of multiple fleet types compared to a single fleet type on the heuristic solution quality and run time. Third, we examine the effect of multiple hubs.

Tables 6.6, 6.7, 6.8 and 6.9 summarize our computational results for varying congestion levels, numbers of fleets, and numbers of hubs. The numbers in parentheses represent computational run times in seconds. To obtain an IP solution, we stopped the solution algorithm either at the 5th feasible IP solution or the 1000th second of run time, whichever came first. Numbers marked with an (*) represent feasible, not optimal, IP solutions.

Table 6.6 considers three hubs and two fleet types with varying levels of demand. As expected, the overall run time increases with demand, reflecting the increasing difficulty of allocating flows to routes. The cutset inequalities are essentially unnecessary when the demand level is quite low (such as 50% of the original demand level) since they are dominated by the connectivity constraints.

Our heuristic approach is particularly fast to execute compared to the LP solution time for EPSND-Cut. In the case when the demand level is low, our heuristic takes less times to generate a feasible solution than even the time to solve the EPSND LP relaxation.

The drawback to our heuristic approach is that as demand levels become very high and the
network is very congested, the quality of the heuristic solution is hard to evaluate. For example, when total demand is increased three fold over the original demand level, our heuristic solution has an integrality gap of about 30%.

Table 6.7 considers a single hub and a single fleet type with varying levels of demand. As expected, this problem is much easier to solve and has a much smaller duality gap than the 3 hubs-2 fleets case.

In Table 6.8, we evaluate the case of a single hub and two fleet types. Compared to the single hub and single fleet scenario, the solution time for EPSND-Cut-LP is reduced surprisingly. It seems that the greater flexibility in assigning route capacity helps the column generation procedure to converge more quickly compared to the single fleet scenario. Although the integrality gap is increased compared to the single hub and single fleet scenario, the AEPSND-Cut solution is able to generate near-feasible solutions.

Finally, we consider, in Table 6.9, the case of a single fleet and three hubs with varying levels of demand. The run time of EPSND-Cut is increased greatly and the solution quality is degraded compared to the single hub-single fleet case, illustrating the large computational impact of moving from a single hub to multiple hubs. Comparing $Z^*_{AEPSND-Cut}$ and $Z^H_{LP}$, we see that AEPSND-Cut does not perform as well in generating near-feasible solutions to EPSND.

Table 6.10 shows the number of columns and rows generated for each scenario. The number of columns for $Z^*_{EPSND-LP}$ and $Z^*_{EPSND-Cut-LP}$ corresponds to the number of package flow variables generated while the number of columns for $Z^*_{AEPSND-Cut-LP}$ indicates the number of fleet design variables generated. Unexpectedly, the number of package flow variables generated in the single fleet-single hub case exceeds those of other scenarios. It appears that providing a richer set of routing capacity assignments, using multiple fleet types, reduces the total number of columns and rows generated in the optimization process and results in comparable (or even faster) overall run times.

Table 6.11 summarizes the number of aircraft used of each fleet type and the percentage of demand passing through each hub for each scenario. Hub 1 is the major hub, to which each location must be connected with at least one route. Fleet type 1 is smaller aircraft, with
<table>
<thead>
<tr>
<th></th>
<th>50 %</th>
<th>100 %</th>
<th>150 %</th>
<th>200 %</th>
<th>250 %</th>
<th>300 %</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z^*_{EPSND-LP}$</td>
<td>166,437</td>
<td>166,553</td>
<td>167,449</td>
<td>171,254</td>
<td>183,145</td>
<td>200,579</td>
<td>175,903</td>
</tr>
<tr>
<td></td>
<td>(60)</td>
<td>(82)</td>
<td>(155)</td>
<td>(212)</td>
<td>(188)</td>
<td>(206)</td>
<td>(151)</td>
</tr>
<tr>
<td>$Z^*_{EPSND-Cut-LP}$</td>
<td>166,437</td>
<td>168,785</td>
<td>178,086</td>
<td>189,472</td>
<td>210,590</td>
<td>242,934</td>
<td>192,717</td>
</tr>
<tr>
<td></td>
<td>(987)</td>
<td>(4,580)</td>
<td>(5,735)</td>
<td>(12,131)</td>
<td>(14,394)</td>
<td>(10,347)</td>
<td>(8,029)</td>
</tr>
<tr>
<td>$Z^*_{AEPSND-Cut-LP}$</td>
<td>166,437</td>
<td>168,785</td>
<td>178,086</td>
<td>189,362</td>
<td>210,289</td>
<td>242,471</td>
<td>192,572</td>
</tr>
<tr>
<td></td>
<td>(7)</td>
<td>(12)</td>
<td>(30)</td>
<td>(46)</td>
<td>(74)</td>
<td>(57)</td>
<td>(38)</td>
</tr>
<tr>
<td>$Z^*_{AEPSND-Cut}$</td>
<td>166,632</td>
<td>169,631</td>
<td>184,629</td>
<td>216,908</td>
<td>245,179</td>
<td>273,202</td>
<td>209,364</td>
</tr>
<tr>
<td></td>
<td>(5)</td>
<td>(6)</td>
<td>(35)</td>
<td>(41)</td>
<td>(74)</td>
<td>(106)</td>
<td>(45)</td>
</tr>
<tr>
<td>$Z^*_{LP}$</td>
<td>166,632</td>
<td>169,631</td>
<td>189,224</td>
<td>226,631</td>
<td>251,108</td>
<td>285,392</td>
<td>214,770</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(2)</td>
<td>(3)</td>
<td>(3)</td>
<td>(2)</td>
</tr>
<tr>
<td>$Z^*_{LP}$</td>
<td>166,632</td>
<td>169,576*</td>
<td>233,970*</td>
<td>258,034*</td>
<td>317,447*</td>
<td>322,548</td>
<td>222,548</td>
</tr>
<tr>
<td></td>
<td>(2)</td>
<td>(3)</td>
<td>(180)</td>
<td>(285)</td>
<td>(310)</td>
<td>(130)</td>
<td>(222)</td>
</tr>
</tbody>
</table>

Table 6.6: Scenario Analysis: Two Fleet Types and Three Hubs

<table>
<thead>
<tr>
<th></th>
<th>50 %</th>
<th>100 %</th>
<th>150 %</th>
<th>200 %</th>
<th>250 %</th>
<th>300 %</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z^*_{EPSND-LP}$</td>
<td>166,437</td>
<td>167,790</td>
<td>174,330</td>
<td>190,871</td>
<td>211,904</td>
<td>236,857</td>
<td>191,365</td>
</tr>
<tr>
<td></td>
<td>(7)</td>
<td>(15)</td>
<td>(23)</td>
<td>(20)</td>
<td>(15)</td>
<td>(14)</td>
<td>(16)</td>
</tr>
<tr>
<td>$Z^*_{EPSND-Cut-LP}$</td>
<td>166,437</td>
<td>174,351</td>
<td>187,922</td>
<td>216,056</td>
<td>240,676</td>
<td>278,492</td>
<td>210,656</td>
</tr>
<tr>
<td></td>
<td>(11)</td>
<td>(50)</td>
<td>(602)</td>
<td>(571)</td>
<td>(1,431)</td>
<td>(1,008)</td>
<td>(612)</td>
</tr>
<tr>
<td>$Z^*_{AEPSND-Cut-LP}$</td>
<td>166,437</td>
<td>174,351</td>
<td>187,877</td>
<td>215,572</td>
<td>240,123</td>
<td>274,649</td>
<td>209,835</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(3)</td>
<td>(14)</td>
<td>(12)</td>
<td>(20)</td>
<td>(18)</td>
<td>(11)</td>
</tr>
<tr>
<td>$Z^*_{AEPSND-Cut}$</td>
<td>167,362</td>
<td>174,604</td>
<td>192,832</td>
<td>219,176</td>
<td>248,981</td>
<td>288,066</td>
<td>215,170</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(6)</td>
<td>(19)</td>
<td>(25)</td>
<td>(72)</td>
<td>(21)</td>
</tr>
<tr>
<td>$Z^*_{LP}$</td>
<td>167,362</td>
<td>183,202</td>
<td>199,490</td>
<td>229,611</td>
<td>258,966</td>
<td>292,819</td>
<td>221,908</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(2)</td>
<td>(1)</td>
</tr>
<tr>
<td>$Z^*_{LP}$</td>
<td>167,362</td>
<td>183,202</td>
<td>199,490</td>
<td>231,653</td>
<td>268,009</td>
<td>300,667</td>
<td>225,064</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(70)</td>
<td>(100)</td>
<td>(29)</td>
</tr>
</tbody>
</table>

Table 6.7: Scenario Analysis: Single Fleet Type and Single Hub

102
### Table 6.8: Scenario Analysis: Two Fleet Types and Single Hub

<table>
<thead>
<tr>
<th></th>
<th>50 %</th>
<th>100 %</th>
<th>150 %</th>
<th>200 %</th>
<th>250 %</th>
<th>300 %</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z_{EPSND-LP}$</td>
<td>166,442</td>
<td>167,062</td>
<td>169,915</td>
<td>182,226</td>
<td>202,890</td>
<td>227,780</td>
<td>186,053</td>
</tr>
<tr>
<td>$Z_{EPSND-Cut-LP}$</td>
<td>166,462</td>
<td>168,788</td>
<td>178,202</td>
<td>193,594</td>
<td>217,532</td>
<td>253,582</td>
<td>196,360</td>
</tr>
<tr>
<td></td>
<td>(19)</td>
<td>(288)</td>
<td>(291)</td>
<td>(197)</td>
<td>(661)</td>
<td>(1,191)</td>
<td>(441)</td>
</tr>
<tr>
<td>$Z_{AEPSND-Cut-LP}$</td>
<td>166,437</td>
<td>168,785</td>
<td>178,086</td>
<td>190,430</td>
<td>214,173</td>
<td>247,983</td>
<td>194,816</td>
</tr>
<tr>
<td></td>
<td>(3)</td>
<td>(5)</td>
<td>(12)</td>
<td>(14)</td>
<td>(30)</td>
<td>(27)</td>
<td>(15)</td>
</tr>
<tr>
<td>$Z_{AEPSND-Cut}$</td>
<td>167,364</td>
<td>170,211</td>
<td>189,384</td>
<td>214,318</td>
<td>235,341</td>
<td>279,642</td>
<td>209,377</td>
</tr>
<tr>
<td></td>
<td>(2)</td>
<td>(3)</td>
<td>(20)</td>
<td>(18)</td>
<td>(70)</td>
<td>(46)</td>
<td>(27)</td>
</tr>
<tr>
<td>$Z_{HP}$</td>
<td>167,364</td>
<td>170,211</td>
<td>196,336</td>
<td>227,440</td>
<td>240,089</td>
<td>277,577</td>
<td>213,170</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(3)</td>
<td>(1)</td>
<td>(2)</td>
<td>(2)</td>
<td>(3)</td>
<td>(2)</td>
</tr>
<tr>
<td>$Z_{EPSND-LP}$</td>
<td>167,364</td>
<td>170,211</td>
<td>196,671</td>
<td>233,607</td>
<td>241,979</td>
<td>294,055*</td>
<td>217,315</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(59)</td>
<td>(190)</td>
<td>(1,000)*</td>
<td>(209)</td>
<td></td>
</tr>
<tr>
<td>$Z_{EPSND-Cut-LP}$ vs. $Z_{EPSND-Cut-LP}$</td>
<td>0.01%</td>
<td>1.03%</td>
<td>4.88%</td>
<td>6.24%</td>
<td>7.22%</td>
<td>11.33%</td>
<td>5.54%</td>
</tr>
<tr>
<td>$Z_{AEPSND-Cut}$ vs. $Z_{HP}$</td>
<td>0.00%</td>
<td>0.00%</td>
<td>3.67%</td>
<td>6.12%</td>
<td>2.02%</td>
<td>-0.74%</td>
<td>1.85%</td>
</tr>
<tr>
<td>$Z_{EPSND-Cut-LP}$ vs. $Z_{HP}$</td>
<td>0.54%</td>
<td>0.84%</td>
<td>10.36%</td>
<td>20.67%</td>
<td>11.24%</td>
<td>15.96%</td>
<td>9.94%</td>
</tr>
</tbody>
</table>

### Table 6.9: Scenario Analysis: Single Fleet Type and Three Hubs

<table>
<thead>
<tr>
<th></th>
<th>50 %</th>
<th>100 %</th>
<th>150 %</th>
<th>200 %</th>
<th>250 %</th>
<th>300 %</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Z_{EPSND-LP}$</td>
<td>166,437</td>
<td>166,554</td>
<td>168,277</td>
<td>176,775</td>
<td>190,808</td>
<td>208,669</td>
<td>179,587</td>
</tr>
<tr>
<td></td>
<td>(10)</td>
<td>(17)</td>
<td>(39)</td>
<td>(48)</td>
<td>(61)</td>
<td>(59)</td>
<td>(39)</td>
</tr>
<tr>
<td>$Z_{EPSND-Cut-LP}$</td>
<td>166,437</td>
<td>174,351</td>
<td>186,825</td>
<td>212,970</td>
<td>235,093</td>
<td>267,480</td>
<td>207,193</td>
</tr>
<tr>
<td></td>
<td>(18)</td>
<td>(245)</td>
<td>(1,843)</td>
<td>(1,585)</td>
<td>(5,033)</td>
<td>(3,482)</td>
<td>(2,034)</td>
</tr>
<tr>
<td>$Z_{AEPSND-Cut-LP}$</td>
<td>166,437</td>
<td>174,351</td>
<td>186,825</td>
<td>212,889</td>
<td>234,946</td>
<td>267,042</td>
<td>207,083</td>
</tr>
<tr>
<td></td>
<td>(5)</td>
<td>(8)</td>
<td>(35)</td>
<td>(38)</td>
<td>(38)</td>
<td>(36)</td>
<td>(27)</td>
</tr>
<tr>
<td>$Z_{AEPSND-Cut}$</td>
<td>166,632</td>
<td>174,929</td>
<td>189,376</td>
<td>213,542</td>
<td>246,977</td>
<td>275,657</td>
<td>211,186</td>
</tr>
<tr>
<td></td>
<td>(3)</td>
<td>(3)</td>
<td>(13)</td>
<td>(31)</td>
<td>(60)</td>
<td>(112)</td>
<td>(37)</td>
</tr>
<tr>
<td>$Z_{HP}$</td>
<td>166,632</td>
<td>181,674</td>
<td>196,250</td>
<td>218,682</td>
<td>256,543</td>
<td>283,926</td>
<td>217,285</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(4)</td>
<td>(2)</td>
</tr>
<tr>
<td>$Z_{EPSND-LP}$</td>
<td>166,632</td>
<td>181,674</td>
<td>210,049</td>
<td>230,874</td>
<td>273,521</td>
<td>309,343*</td>
<td>228,682</td>
</tr>
<tr>
<td></td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(30)</td>
<td>(1,000)*</td>
<td>(172)</td>
</tr>
<tr>
<td>$Z_{EPSND-Cut-LP}$ vs. $Z_{EPSND-Cut-LP}$</td>
<td>0.00%</td>
<td>4.68%</td>
<td>11.02%</td>
<td>20.48%</td>
<td>23.21%</td>
<td>28.18%</td>
<td>15.37%</td>
</tr>
<tr>
<td>$Z_{AEPSND-Cut}$ vs. $Z_{HP}$</td>
<td>0.00%</td>
<td>3.86%</td>
<td>3.63%</td>
<td>2.41%</td>
<td>3.87%</td>
<td>3.00%</td>
<td>2.79%</td>
</tr>
<tr>
<td>$Z_{EPSND-Cut-LP}$ vs. $Z_{HP}$</td>
<td>0.12%</td>
<td>4.20%</td>
<td>12.43%</td>
<td>8.41%</td>
<td>16.35%</td>
<td>15.65%</td>
<td>9.53%</td>
</tr>
</tbody>
</table>

103
capacity equal to $1/2$ that of fleet type 2. Fleet type 2 has only two aircraft. In the single hub scenarios, less aircraft are used than in the multiple hub scenarios. Also, when two fleets are available, fewer aircraft are needed than when there is only one fleet type.

The results of our scenario analyses are summarized as:

- Run time and quality of solution deteriorates as demand increases. Excluding cutset inequalities has an increasingly negative impact as demand increases.

- Near optimal solutions are found when the demand level is low, largely because connectivity constraints are the only tight constraints.

- An increase in the number of fleets results in improved solution quality and surprisingly comparable run times.

- The minimum number of aircraft used occurs for the single hub networks, since maximum consolidation of demand is possible. In the multiple hub networks, although the number of aircraft used increases, the variable operating costs decrease and service reliability increases (reduced total distance traveled and more capacity in the system).
<table>
<thead>
<tr>
<th>Scenario</th>
<th>Model</th>
<th># cols</th>
<th>50%</th>
<th>100%</th>
<th>150%</th>
<th>200%</th>
<th>250%</th>
<th>300%</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Fleet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Hub</td>
<td>$Z^{*}_{EPSND-LP}$</td>
<td># cols</td>
<td>720</td>
<td>912</td>
<td>984</td>
<td>1,128</td>
<td>1,128</td>
<td>1,080</td>
<td>992</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>22</td>
<td>240</td>
<td>366</td>
<td>496</td>
<td>376</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{EPSND-Cut-LP}$</td>
<td># cols</td>
<td>693</td>
<td>1,480</td>
<td>3,696</td>
<td>3,860</td>
<td>4,968</td>
<td>4,896</td>
<td>3,202</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>12</td>
<td>248</td>
<td>322</td>
<td>562</td>
<td>496</td>
<td>273</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{AEPSND-Cut-LP}$</td>
<td># cols</td>
<td>838</td>
<td>738</td>
<td>750</td>
<td>842</td>
<td>845</td>
<td>840</td>
<td>809</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>12</td>
<td>248</td>
<td>322</td>
<td>562</td>
<td>496</td>
<td>273</td>
</tr>
<tr>
<td>Single Fleet</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Three Hubs</td>
<td>$Z^{*}_{EPSND-LP}$</td>
<td># cols</td>
<td>468</td>
<td>600</td>
<td>792</td>
<td>936</td>
<td>1,032</td>
<td>1,056</td>
<td>814</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>12</td>
<td>234</td>
<td>314</td>
<td>526</td>
<td>386</td>
<td>229</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{EPSND-Cut-LP}$</td>
<td># cols</td>
<td>491</td>
<td>1,190</td>
<td>3,119</td>
<td>3,048</td>
<td>4,272</td>
<td>3,936</td>
<td>2,676</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>20</td>
<td>226</td>
<td>290</td>
<td>430</td>
<td>378</td>
<td>224</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{AEPSND-Cut-LP}$</td>
<td># cols</td>
<td>1,805</td>
<td>1,785</td>
<td>1,825</td>
<td>1,803</td>
<td>1,806</td>
<td>1,702</td>
<td>1,788</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>12</td>
<td>234</td>
<td>314</td>
<td>526</td>
<td>386</td>
<td>229</td>
</tr>
<tr>
<td>Two Fleets</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Single Hub</td>
<td>$Z^{*}_{EPSND-LP}$</td>
<td># cols</td>
<td>621</td>
<td>576</td>
<td>672</td>
<td>1,080</td>
<td>1,080</td>
<td>912</td>
<td>824</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>12</td>
<td>58</td>
<td>72</td>
<td>210</td>
<td>202</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{EPSND-Cut-LP}$</td>
<td># cols</td>
<td>456</td>
<td>1,632</td>
<td>1,700</td>
<td>1,824</td>
<td>3,336</td>
<td>4,128</td>
<td>2,179</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>12</td>
<td>58</td>
<td>72</td>
<td>210</td>
<td>202</td>
<td>92</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{AEPSND-Cut-LP}$</td>
<td># cols</td>
<td>967</td>
<td>1,303</td>
<td>1,437</td>
<td>1,044</td>
<td>1,336</td>
<td>1,139</td>
<td>1,204</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>10</td>
<td>77</td>
<td>206</td>
<td>477</td>
<td>375</td>
<td>191</td>
</tr>
<tr>
<td>Two Fleets</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Three Hubs</td>
<td>$Z^{*}_{EPSND-LP}$</td>
<td># cols</td>
<td>404</td>
<td>454</td>
<td>575</td>
<td>696</td>
<td>816</td>
<td>1,008</td>
<td>659</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>11</td>
<td>59</td>
<td>181</td>
<td>377</td>
<td>244</td>
<td>145</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{EPSND-Cut-LP}$</td>
<td># cols</td>
<td>456</td>
<td>1,064</td>
<td>1,934</td>
<td>3,096</td>
<td>4,344</td>
<td>4,320</td>
<td>2,536</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>11</td>
<td>59</td>
<td>181</td>
<td>377</td>
<td>244</td>
<td>145</td>
</tr>
<tr>
<td></td>
<td>$Z^{*}_{AEPSND-Cut-LP}$</td>
<td># cols</td>
<td>2,266</td>
<td>2,290</td>
<td>2,609</td>
<td>2,655</td>
<td>2,662</td>
<td>2,718</td>
<td>2,533</td>
</tr>
<tr>
<td></td>
<td></td>
<td># rows</td>
<td>0</td>
<td>10</td>
<td>77</td>
<td>201</td>
<td>455</td>
<td>295</td>
<td>173</td>
</tr>
</tbody>
</table>

Table 6.10: Scenario Analysis: Number of Columns, Rows Generated
<table>
<thead>
<tr>
<th></th>
<th>% of Demand using Hub</th>
<th>50 %</th>
<th>100 %</th>
<th>150 %</th>
<th>200 %</th>
<th>250 %</th>
<th>300 %</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Single Fleet</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Single Hub</strong></td>
<td>hub 1</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td></td>
<td>hub 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>hub 3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Aircraft used</strong></td>
<td>type 1</td>
<td>12</td>
<td>13</td>
<td>16</td>
<td>20</td>
<td>23</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>type 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Single Fleet</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Three Hubs</strong></td>
<td>hub 1</td>
<td>99.74%</td>
<td>91.52%</td>
<td>93.90%</td>
<td>89.97%</td>
<td>84.38%</td>
<td>83.21%</td>
</tr>
<tr>
<td></td>
<td>hub 2</td>
<td>0.26%</td>
<td>0.26%</td>
<td>2.98%</td>
<td>1.74%</td>
<td>6.06%</td>
<td>7.02%</td>
</tr>
<tr>
<td></td>
<td>hub 3</td>
<td>0.00%</td>
<td>8.22%</td>
<td>3.12%</td>
<td>8.29%</td>
<td>9.56%</td>
<td>9.77%</td>
</tr>
<tr>
<td><strong>Aircraft used</strong></td>
<td>type 1</td>
<td>12</td>
<td>13</td>
<td>18</td>
<td>21</td>
<td>25</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>type 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Two Fleets</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Single Hub</strong></td>
<td>hub 1</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td></td>
<td>hub 2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>hub 3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>Aircraft used</strong></td>
<td>type 1</td>
<td>12</td>
<td>11</td>
<td>12</td>
<td>16</td>
<td>19</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>type 2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td><strong>Two Fleets</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Three Hubs</strong></td>
<td>hub 1</td>
<td>99.74%</td>
<td>99.74%</td>
<td>99.19%</td>
<td>90.55%</td>
<td>82.58%</td>
<td>81.09%</td>
</tr>
<tr>
<td></td>
<td>hub 2</td>
<td>0.26%</td>
<td>0.26%</td>
<td>0.64%</td>
<td>3.83%</td>
<td>6.63%</td>
<td>4.15%</td>
</tr>
<tr>
<td></td>
<td>hub 3</td>
<td>0.00%</td>
<td>0.00%</td>
<td>0.17%</td>
<td>5.62%</td>
<td>10.78%</td>
<td>14.76%</td>
</tr>
<tr>
<td><strong>Aircraft used</strong></td>
<td>type 1</td>
<td>12</td>
<td>11</td>
<td>12</td>
<td>17</td>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>type 2</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 6.11: Scenario Analysis: Percentage of Demand Through Each Hub, Number of Aircraft Used
Chapter 7

Contributions and Future Directions

7.1 Contributions

Our research contributions can be summarized as:

- **Modeling Contributions:**

  We have developed a representative, solvable model for large scale transportation service network design problems with time windows. We present three different but equivalent service network design models: node-arc (or link) formulation, path formulation and tree formulation. With the use of *route based decision variables*, we capture complex cost structures (i.e., non-linear and flow-dependent link or route costs) and satisfy implicitly very complicated rules. The use of route based decision variables helps us reduce the number of constraints in the service network design problem formulations. For example, in the case of our package flow problem, we reduce the number of constraints in the node-arc formulation from 14,155,336 to 17,832 using the path formulation and to 428 using the tree formulation. Direct solution of the link formulation for this application was impossible due to the enormous computational hardware requirements; however, we obtained the solution to the path formulation within 40 minutes and solved the tree formulation within 1 minute.

  By exploiting special problem structure of multimodal express shipment operations, we were able to achieve a dramatic decrease in problem size. Using the *time-line network* representation instead of a conventional time-space network and applying a series of novel problem reduction
methods; namely, derived schedule network, node consolidation and link consolidation, we reduce network size from hundreds of thousands of nodes and links to a reduced-size network containing only 2,400 nodes and 72,144 links.

These reduction techniques transform the problem from impossible to solve to solvable, and do so without losing exactness of the model.

- Algorithmic and Implementation Contributions:

We have developed an algorithm and implementation that finds quality solutions to service network design problems containing hundreds of thousands of constraints and billions of variables. Our optimization approach synthesizes column and row generation techniques. Column generation overcomes the difficulties of a huge number of decision variables. Row generation allows us to overcome the difficulties associated with a weak LP relaxation, namely, the need to generate potentially many valid inequalities. By generating columns and rows on an “as-needed” basis, we solve service network design problems containing only a small fraction of the constraint matrix.

As a proof-of-concept, we applied our models and solution algorithms to a multimodal express package shipment application. Although we were unable to generate optimal IP solutions, partly due to the huge application size (even after a series of novel problem reduction methods) and the NP-complete classification of the service network design problems, our heuristic approach provides quality feasible solutions and results in annual operating cost savings of tens of millions of dollars and a reduction in the fleet size required, within acceptable run times.

- Practical Contributions:

Our service network design models and solution algorithms apply to service network design problems arising at railroads, airlines, trucking firms, intermodal partnerships, etc. All of these problems require the determination of a cost minimizing or profit maximizing set of services and their schedules, given resource constraints. Depending on the application, additional side constraints required for our express package application such as fleet balance constraints, fleet size constraints, hub sort capacity, hub landing capacity and connectivity constraints, can be either relaxed or viewed differently. For example, in many freight transportation-related service
<table>
<thead>
<tr>
<th>Challenges</th>
<th>Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem size</td>
<td>Dynamic column and cut generation</td>
</tr>
<tr>
<td>time and space</td>
<td>network reduction methods</td>
</tr>
<tr>
<td>near complete demand pattern</td>
<td>tree multicommodity formulation</td>
</tr>
<tr>
<td>Complex cost structures and rules</td>
<td>Route-based decision variables</td>
</tr>
<tr>
<td>fixed costs, route-based costs</td>
<td>allocate fixed costs to integer variables</td>
</tr>
<tr>
<td>Interdependencies between design variables</td>
<td>add fleet size constraints</td>
</tr>
<tr>
<td>fleet size</td>
<td>add fleet balance constraints</td>
</tr>
<tr>
<td>fleet balance requirements</td>
<td>Add valid inequalities</td>
</tr>
<tr>
<td>Weak LP relaxations</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.1: Challenges and Resolution

network design problems, we may relax connectivity constraints. Hub sort capacity constraints can be viewed as capacity constraints on distribution centers (depots, warehouses, etc.) and hub landing capacity can be thought of as limits on the number of truck arrivals to distribution centers due to limited resources at the docking facility.

Our solution approach for service network design will shorten current manual, lengthy planning processes and give planners more time in developing scenarios, analyzing the quality of solutions and performing strategic planning.

Table 7.1 summarizes the research challenges and resolutions of modeling and solving the express package service network design problem.

### 7.2 Future Directions

The first and immediate task is to validate and evaluate more thoroughly our models and their solution quality. Although we validated our solution quality, true quantification of benefits from our models must be achieved.

Second, we need to develop an efficient and tractable branching strategy for use within a branch-and-price solution algorithm. Although our heuristic approach finds an optimal solution to the LP relaxation using column and row generation, we obtain an IP solution using a branch-and-bound solution procedure that does not involve further column or row generation.

Third, we limit our cutset generation due to the enormous memory requirements and com-
putational expense of our explicit cutset generation procedure. Finding an efficient, or at least relatively quick, separation algorithm for implicit generation of aggregate capacity demand inequalities (plus lifted ones either by C-G cuts or cutset inequalities) would reduce a lot of the computational burden. It is probably worthwhile to develop good approximate separation algorithms if an exact algorithm is computationally expensive. Improved LP solution quality allows us to generate much better service network design solutions.
Appendix A

Service Design Problem Notations

A.1 Sets

\( N \) : node set

\( A \) : arc sets

\( F(\exists f) \) : set of service types

\( R^f(\exists r) \) : set of design sets for a service type \( f \in F \)

\( K(\exists k) \) : Set of commodities, with each \( k \in K \) defined by an origin \( O(k) \) and a destination \( D(k) \)

\( P^k(\exists p) \) : set of paths from \( O(k) \) to \( D(k) \) for each \( k \in K \)

\( s_i \) : \( = \{k| O(k) = i, k \in K\} \), a super-commodity comprised of all commodities \( k \in K \) originating at \( i \)

\( S(\exists s_i) \) : \( = \{s_i| i \in O(k), k \in K\} \), set of super-commodities \( s_i, i \in O(k) \), for all \( k \in K \)

\( q \) : \( = \{(i,j)| (i,j) \in p^k_q, p^k_q \in P^k, k \in s_{O(k)}\} \), a tree comprised of paths \( p^k_q \), \( p^k_q \in P^k, k \in s_{O(k)} \)

\( p^k_q \) : a path from \( O(k) \) to \( D(k) \) in tree \( q \)

\( Q^{s_{O(k)}}(\exists q) \) : set of trees originating at \( O(k) \), for all super commodities \( s_{O(k)} \in S \)
A.2 Decision Variables

\[ y^f_r : \text{design route flow of } r \in R^f, \text{ for } f \in F \]
\[ y^f_{ij} : \text{design link flow on arc } (i, j) \in A, \text{ for } f \in F \]
\[ x^k_p : \text{fraction of } b^k \text{ on path } p \in P^k \text{ for } k \in K \]
\[ x^k_{ij} : \text{fraction of } b^k \text{ on arc } (i, j) \in A, \text{ for } k \in K \]
\[ w^s_{O(k)} : \text{fraction of } b^k \text{ assigned to path } p^k_q \text{ for each } k \in s_{O(k)}, \text{ for all } q \in Q^{s_{O(k)}} \text{ and all } s_{O(k)} \in S \]

A.3 Parameters

\[ b^k : \text{total quantity of commodity } k \in K \]
\[ u_{ij} : \text{capacity of arc } (i, j) \in A \]
\[ u^f : \text{capacity of a service type } f \in F \]
\[ n^f : \text{fleet size of a service type } f \in F \]
\[ h^f_r : \text{fixed cost of providing one unit of design set of service type } f \in F \text{ along route } r \in R^f \]
\[ h^f_{ij} : \text{fixed cost of providing one unit of design set of service type } f \in F \text{ along arc } (i, j) \in A \]
\[ c^k_p : \text{cost of sending one unit of flow of } k \in K \text{ along path } p \in P^k \]
\[ c^k_{ij} : \text{cost of sending one unit of commodity } k \in K \text{ on arc } (i, j) \in A \]
\[ c^s_{O(k)} : \text{total cost of sending } b^k \text{ units of } k \in s_{O(k)} \text{ along the path } p^k_q, \text{ for all } q \in Q^{s_{O(k)}} \text{ and all } s_{O(k)} \in S \]
A.4 Indicators

\[ \alpha_{ij}^r : \begin{cases} 
1 & \text{if arc } (i, j) \in A \text{ belongs to } r, \text{ where } r \in R^f, \ f \in F \\
0 & \text{otherwise} 
\end{cases} \]

\[ \delta_{ij}^p : \begin{cases} 
1 & \text{if arc } (i, j) \in A \text{ belongs to } p, \text{ where } p \in P^k, \ k \in K \\
0 & \text{otherwise} \\
-1 & \text{if a node } i \in N \text{ is the start node of } r 
\end{cases} \]

\[ \beta_i^r : \begin{cases} 
1 & \text{if a node } i \in N \text{ is the end node of } r, \text{ where } r \in R^f, \ f \in F \\
0 & \text{otherwise} 
\end{cases} \]

\[ \Gamma_{ij}^q : \begin{cases} 
1 & \text{if arc } (i, j) \in A \text{ belongs to } q, \text{ where } q \in Q^s, \ s \in S \\
0 & \text{otherwise} 
\end{cases} \]
Bibliography


[65] Savelsbergh, M.W.P. *A Branch-and-Price Algorithm for the Generalized Assignment Problem*, Report COC-9302, Georgia Institute of Technology, Atlanta, Georgia


