Two-Stage Service Provision by Branch and Bound

Shane Dye
Department of Management
University of Canterbury
Christchurch, New Zealand
s.dye@mang.canterbury.ac.nz

Asgeir Tomasgard
SINTEF, Trondheim, Norway

Stein W. Wallace
NTNU, Trondheim, Norway

Abstract

Bender's decomposition is often used to solve two-stage stochastic programming problems. The problem is transformed into a master problem and many subproblems with similar structure. For a two-stage problem with integer variables in the first-stage branch and bound may be used to resolve the integral requirements. This involves solving many relaxations of the problem. Using Bender's decomposition to solve these relaxations means there are many, many subproblems to solve with similar structure.

This paper considers such problems where the subproblems have a network or transportation problem structure. Ways of exploiting the structure of the subproblems and the combination of branch and bound with Bender's decomposition are discussed.

1 Introduction

The telecommunication service provision problem may be modelled as a two-stage stochastic programming problem with a mixed integer first stage, Tomasgard *et al.* [6]. When the first stage decisions are fixed the problem naturally decomposes into many network problems with a transportation structure. Background, modelling issues and solution approaches for the service provision problem are discussed by Tomasgard [5].

This paper looks at a particular solution approach to the deterministic equivalent problem. Branch and bound is used with Bender's decomposition for the bounds at the branch and bound nodes. The structure of the subproblems is exploited through preprocessing. Management of these Bender's decomposition through the branch and bound tree is also discussed.

An underlying intention of this paper is to gain insight into the problem through the development of the algorithm.

2 The Service Provision Problem

The formulation given here can also be found in Tomasgard *et al.* [6]. That paper also details the background of the model.

A service provider has m computers available to provide processing services for n different services. The computers have processing capacities s_i, $i = 1, \ldots, m$. The service provider must decide which computers to install which services on in order to best meet future demands. Service j uses r_j units of processing capacity, $j = 1, \ldots, n$, to be available for use on a computer, in addition to those used serving customers. Demand that is not served must be met externally at a cost of q_j per unit, $j = 1, \ldots, n$. The installation decision is made before demand is known with certainty. This is modelled as binary decision variable z_{ij}, where $z_{ij} = 1$ indicates that service j is installed on computer i.

The deterministic equivalent uses ℓ scenarios for future demand. Scenario k, $k = 1, \ldots, \ell$, occurs with probability p_k and has a demand of δ_{jk} processing units for service j, $j = 1, \ldots, n$. In scenario k, for service j decision variable x_{ijk} indicates how much of the processing demand is met by computer i and x_{0jk} indicates how much of this met externally. Demand for service j can only be met by a computer with service j installed.

The objective is to minimize the expected cost of externally met demand. The deterministic equivalent is as follows.

$$
\begin{array}{lrcll}
\min & \displaystyle\sum_{k=1}^{\ell} p_k \sum_{j=1}^{n} q_j x_{0jk} & & & \\
\text{s.t.} & \displaystyle\sum_{j=1}^{n} r_j z_{ij} \quad + \sum_{j=1}^{n} x_{ijk} & \leq & s_i & i = 1, ..., m,\ k = 1, ..., \ell \\
& x_{0jk} + \displaystyle\sum_{i=1}^{m} x_{ijk} & = & \delta_{jk} & j = 1, ..., n,\ k = 1, ..., \ell \\
& M_{ijk} z_{ij} \quad - x_{ijk} & \geq & 0 & i = 1, ..., m,\ j = 1, ..., n,\ k = 1, ..., \ell \\
& z_{ij} \in \{0, 1\}, \quad x_{ijk} & \geq & 0 & i = 1, ..., m,\ j = 1, ..., n,\ k = 1, ..., \ell
\end{array}
\tag{1}
$$

Notice that when the first-stage variables are fixed the problem decomposes into ℓ independent transportation problems.

The variable upper bound on x_{ijk}, $M_{ijk} z_{ij}$, is to require that x_{ijk} is only positive when $z_{ij} = 1$. Constant M_{ijk} is required to be an upper bound on x_{ijk}. A tight choice is $M_{ijk} = \min\{\delta_{jk}, \max\{s_i - r_j, 0\}\}$.

(1) is simply a mixed integer program and can be solved directly by any mixed integer solution technique. The potential difficulty here is the large problem size as the number of scenario's gets large.

3 Branch and Bound with Bender's

One method for solving mixed integer programs is branch and bound. For a general discussion of branch and bound see, for example, Nemhauser and Wolsey [4]. The general method is to use a relaxation of the problem to find a lower bound on the solution value, divide the feasible region into smaller subregions and apply the

method recursively to each of these subregions. The lower bound generation and feasible region partitioning need to be conducted in such a way as to ensure that after a finite number of branches the lower bound is tight and a feasible solution is generated. Subregions are pruned when they are empty or the lower bound indicates that the subregion does not contain an optimal solution.

The most common implementation is to use a linear program relaxation for computing the lower bounds. In this case the linear relaxation is a deterministic equivalent of a two-stage stochastic *linear* program with recourse. It has a lot of special structure and might be very large.

Instead of tackling the deterministic equivalent directly, any method for solving two-stage stochastic linear programs may be used to obtain these lower bounds. One possibility is to use Bender's decomposition (the L-shaped method). Here part of the objective function is approximated using dual solutions from second-stage problems. For a general discussion of Bender's decomposition see Nemhauser and Wolsey [4]. For a discussion of its use in stochastic programming see Wallace and Kall [2].

Bender's decomposition divides a mathematical program into two parts, a master problem and a subproblem. The original formulation is projected onto the master problem variables with the projection defined by dual information from the subproblem. The projection can be achieved by sequentially adding two types of extra constraints to the master problem: optimality cuts and feasibility cuts. Optimality cuts define how the subproblem's objective function affects the master problem objective function. An additional variable, θ, is used to track the effect. Feasibility cuts define how the subproblem's constraints affects the master problem feasible region. Not all cuts are needed, only those that serve to define and prove optimality of the full problem. Indeed, some early cuts may be removed.

Using Bender's decomposition at the branch and bound nodes leads to a wide choice in implementation. Four main issues are: the formulation of the master problem, Bender's cut generation, how long to spend generating cuts, and cut management throughout the branch and bound tree. What follows is a brief discussion of these issues.

3.1 Master Problem Formulation

The classical L-shaped form of Bender's decomposition for two-stage stochastic programs, [7], takes the first-stage variables to be in the master problem so that the subproblem naturally decomposes into a separate problem for each scenario. The master problem at iteration ν may be formulated as follows.

$$\begin{aligned} \min \quad & \theta \\ \text{s.t.} \quad & \sum_{j=1}^{n} r_j z_{ij} \le s_i \quad i = 1, ..., m \\ & \theta + \sum_{i=1}^{m} \sum_{j=1}^{n} a^{\nu}_{ijt} z_{ij} \ge b^{\nu}_t \quad t = 1, ..., v^{\nu} \\ & z_{ij} \in \{0, 1\} \qquad i = 1, ..., m,\ j = 1, ..., n \end{aligned} \tag{2}$$

where there are v^{ν} Bender's optimality cuts, A^{ν} is the matrix of their coefficients and b^{ν} the vector of their right hand sides. These coefficients are the expected

value of corresponding coefficients arising from a separate cut for each subproblem individually. The first m cuts are the only important Bender's feasibility cuts, see Dye *et al.* [1].

The full first-stage the objective function encapsulates all information about profitable variable values. When the objective function approximation is coarse most of this information is lost. To provide more information about how different decisions for the master problem can affect the subproblems early in the Bender's process it may be useful to include more information in the master problem. The price for this additional information will be a longer time spent solving the master problem each iteration of the Bender's decomposition.

3.1.1 Special Scenario in the Master

Consider including a single 'special' scenario explicitly as a part of the master problem. Introducing new first-stage scenario 1 variables x^1_{ij}, $i = 0, 1, ..., m$, $j = 1, ..., n$, the master problem (2) becomes:

$$
\begin{array}{lrll}
\min & p_1 \sum_{j=1}^{n} q_j x^1_{0j} + (1 - p_1)\theta & & \\
\text{s.t.} & \sum_{j=1}^{n} r_j z_{ij} + \sum_{j=1}^{n} x^1_{ij} \leq s_i & i = 1, ..., m & \\
& \sum_{i=1}^{m} x^1_{ij} \leq \delta_{j1} & j = 1, ..., n & (3) \\
& M_{ij1} z_{ij} - x^1_{ij} \geq 0 & i = 1, ..., m,\ j = 1, ..., n & \\
& \theta + \sum_{i=1}^{m} \sum_{j=1}^{n} a^{\nu}_{ijt} z_{ij} \geq b^{\nu}_{t} & t = 1, ..., v^{\nu} & \\
& z_{ij} \in \{0, 1\}, \quad x^1_{ij} \geq 0 & i = 0, 1, ..., m,\ j = 1, ..., n &
\end{array}
$$

The choice of the scenario to be special will affect the number of Bender's iterations necessary. If p_1 is small the effect of the special scenario on the choice of first-stage solution may not be worth the extra effort to solve the more complex first-stage problem.

3.1.2 Expected Value Master Problem

The expected value problem (where each uncertain parameter is replaced with its expected value) provides a lower bound on the optimal solution value when the uncertainty is present only in the right hand side parameters [2]. The expected value problem can be incorporated as part of the master problem. The expected value problem may also be thought of as an approximation arising from an explicit initial choice of Bender's cuts. Those cuts which are formed by picking the *same* dual solution for each scenario [2]. Introducing expected value variables $\bar{x}_{ij}$, $i = 0, 1, ..., m$, $j = 1, ..., n$, master problem (2) changes as follows.

$$
\begin{array}{lllll}
\min & \theta & & & \\
\text{s.t.} & \sum_{j=1}^{n} r_j z_{ij} & + \sum_{j=1}^{n} \bar{x}_{ij} \leq s_i & i = 1, ..., m & \\
& & \sum_{i=1}^{m} \bar{x}_{ij} \leq d_j & j = 1, ..., n & \\
& \bar{M}_{ij} z_{ij} & - \bar{x}_{ij} \geq 0 & i = 1, ..., m,\ j = 1, ..., n & (4) \\
& \theta & - \sum_{i=1}^{m} \sum_{j=1}^{n} q_j \bar{x}_{0j} \geq 0 & & \\
& \theta & + \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ijt}^{\nu} z_{ij} \geq b_t^{\nu} & t = 1, ..., v^{\nu} & \\
& z_{ij} \in \{0, 1\}, & \bar{x}_{ij} \geq 0 & i = 0, 1, ..., m,\ j = 1, ..., n &
\end{array}
$$

where $d_j = E_k[\delta_{jk}]$, $\bar{M}_{ij}$ is an upper bound on $\bar{x}_{ij}$, $i = 1, ..., m$, $j = 1, ..., n$. The expected value problem objective function is eventually superseded by the objective function approximation. This may mean it is advantageous to drop the expected value part once sufficient cuts have been generated.

3.2 Cut Generation

One implementation of the L-shaped method is to solve the subproblems directly and individually. Given efficient transportation algorithms this should work well for small ℓ. This could be improved by using bunching [2, Section 3.10]; solving groups of subproblems simultaneously.

Alternatively, preprocessing could be used to reduce the computational time required to solve each subproblem [8]. This, in turn, may lead to a reduced total computational effort. Dye *et al.* [1] characterize all important Bender's cuts arising from each individual subproblem for a general objective function and when $q_j = 1$, $j = 1, \ldots, n$. The cuts described are strong cuts [3] and could be used accelerate the Bender's decomposition.

When $q_j = 1$ for $j = 1, \ldots, n$ and the first stage variables are free the number of cuts that need to be searched is in the order of 2^{n+m}. It is not useful to generate all such cuts *a priori*. When the first stage variables are fixed the set of potentially violated cuts is reduced to less than 2^m. Two schemes are suggested in [1] for generating violated inequalities.

For the general objective function, when the first stage variables are free the number of cuts to be searched is more than $2^n m^{\frac{n}{2}}$. Even when the first stage variables are fixed the number of potentially violated cuts is too large to be usefully searched.

3.3 Bender's Iterations

The question of how many Bender's iterations to perform at each node is important. The more iterations, the tighter the bound and tighter bounds are more likely to fathom the current node. However, if this node will not be fathomed the time spent tightening the bound would be better spent exploring the rest of the branch and

bound tree. Further, with too few iterations the objective function approximation may not provide sufficient information to make a good choice of branching variable.

Preliminary testing suggests that the number of Bender's iterations required to solve each node fully is relatively small. In such a situation it is well worth solving each node to optimality.

Using Bender's decomposition at the branch and bound nodes also reduces the information available for making good branching decisions. This can affect the overall size of the branch and bound tree.

3.4 Cut Management

All cuts generated may be extended to be valid for all feasible values of the first stage variables. This leads to a number of possibilities for reusing previously useful cuts. In each case where cuts are reused, they may either be added to warm start the lower bound approximation or checked during the cut generation process to quickly find violated cuts. In the latter case, however, the violated cuts are unlikely to be the strongest for the subproblem considered. Preliminary testing suggests that time is better spent generating strong cuts than checking stored cuts.

The main difficulty with reusing cuts is cut management. Ensuring the number of stored cuts does not get too large, and finding efficient ways to store and search these cuts.

3.4.1 No Cut Reuse

One possibility is to only use cuts locally. This has the advantages that there is no need to decide which cuts to use at a node and that cuts do not need to be stored between nodes. Preliminary testing suggests that in some instances the time saved by not storing cuts is worthwhile.

3.4.2 Next Node Cut Reuse

Since all cuts are valid throughout the branch and bound tree are one possibility is to reuse the current set of cuts at the next examined node. As the next node can be unrelated to the previously examined node, the current set of cuts might not contain any strong cuts.

3.4.3 Inheriting Cuts

The cuts most likely to be strong for the current branch and bound node are those arising (or inherited) from the parent node. One possibility is to only inherit the cuts that are binding at the final solution in the parent node. Another possibility is to consider the effect of branching on the cuts explicitly, disregarding any cuts that will definitely be weak.

Difficulties that arise from cut inheritance include: communicating stored cuts to child nodes, deciding when to remove unwanted stored cuts and, when memory is at a premium, choosing which cuts to replace. The tradeoff between the time taken to generate cuts and the overhead incurred through storing cuts deserves further investigation.

4 Conclusions and Further Directions

This discussion paper considered the use of Bender's decomposition with branch and bound to solve a two-stage version of the telecommunication service provision problem. Various implementation issues are raised and discussed. These issues need to be tested on realistic data for the problem for their usefulness to be better evaluated. Many of the ideas contained in this paper apply to general two-stage problems with integer first-stage.

It is possible to strengthen the master problem by adding integrality cuts based on the optimality cuts currently included. The cut management routines of Section 3.4 can also be used for integrality cuts.

Acknowledgements

This research was funded in part by Telenor AS, Norway, and in part by the University of Canterbury, New Zealand.

References

[1] Shane Dye, Asgeir Tomasgard, and Stein W. Wallace. Dual extreme points and rays for the servivce provision problem. In preparation, 2000.

[2] P. Kall and S.W. Wallace. *Stochastic Programming*. Wiley, Chichester, 1994.

[3] T. L. Magnanti and R. T. Wong. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29(3):464–484, 1981.

[4] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience series in discrete mathematics and optimization. Wiley, New York, 1988.

[5] A. Tomasgard. *Aspects of Service Provision and Distributed Processing in a Telecommunication Network*. PhD thesis, Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway, October 1998.

[6] Asgeir Tomasgard, Jan A. Audestad, Shane Dye, Leen Stougie, Maarten H. van der Vlerk, and Stein W. Wallace. Modelling aspects of distributed processing in telecommunication networks. *Annals of Operations Research*, 82:161–184, 1998.

[7] R. van Slyke and R. J-B Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal of Applied Mathematics*, 17:638–663, 1969.

[8] S. W. Wallace and R. J-B Wets. Preprocessing in stochastic programming: The case of linear programs. *ORSA Journal on Computing*, 4:45–59, 1992.