Dantzig-Wolfe Decomposition for Solving Multi-Stage Stochastic Capacity-Planning Problems

Kavinesh J. Singh* Andy B. Philpott† R. Kevin Wood‡

July 13, 2005

Abstract

We describe a general multi-stage stochastic integer-programming model for planning discrete capacity expansion of production facilities. A scenario tree represents uncertainty in the model. Variable splitting leads to two forms of this model: the first allows multiple expansions of each facility over the planning horizon while the second allows at most one. Dantzig-Wolfe decomposition of either split-variable model results in a binary master problem that solves easily, as its linear-programming relaxation tends to yield integer solutions. For each scenario-tree node, the decomposition defines a subproblem that may be viewed as a single-period, deterministic capacity-expansion problem. An effective solution procedure results as long as the subproblems solve efficiently, and the procedure incorporates a good "duals stabilization scheme". We present computational results for a model to plan the capacity expansion of a real-world electricity distribution network given uncertain future demand. The largest problem we solve to optimality has 6 stages and 243 scenarios corresponding to a deterministic equivalent with a quarter of a million binary variables.

Key words: Multi-stage stochastic integer program, column generation, branch-and-price, capacity expansion, Dantzig-Wolfe decomposition

Subject classification: Facilities/equipment planning, capacity expansion, discrete, stochastic: multi-stage. Production/scheduling, planning. Programming, stochastic, integer: column generation, Dantzig-Wolfe decomposition, branch-and-price.

1 Introduction

Research from as early as the 1950s (Masse and Gibrat 1957) suggests that effective capacity planning for industrial facilities must treat uncertainty explicitly. The list of uncertain parameters can include demands on those facilities, expansion costs, operating costs, and production efficiencies.

*University of Auckland, Auckland, New Zealand, kavinesh.singh@auckland.ac.nz

†University of Auckland, Auckland, New Zealand, a.philpott@auckland.ac.nz

‡Naval Postgraduate School, Monterey, California, k.wood@nps.edu

This paper studies capacity-planning problems in which a sequence of discrete, capacity-expansion decisions must be made over a finite planning horizon, subject to one or more sources of uncertainty. A deterministic, single-period instance of our model, without capacity-expansion decisions, can be viewed as an operations-planning model for a "system", which might represent a single plant with multiple production facilities, each of which has a fixed production capacity and manufactures multiple products. Given production costs and known product demands, the system manager must identify a minimum-cost, capacity-feasible, operational plan to meet those demands. Even this single-period, deterministic problem can be complicated, as it may require a high level of modeling fidelity that incorporates both continuous and discrete decision variables. However, the full planning problem spans a multi-period horizon, must incorporate capacity-expansion decisions to accommodate demand growth, and faces uncertainty in demand, costs and possibly other parameters. An optimal capacity-expansion plan will (1) enable production to meet demand, and (2) minimize the expected costs of capacity expansion plus production over the planning horizon.

The stochastic capacity-planning problem can be formulated as a multi-stage, stochastic, mixed-integer program that minimizes the expected discounted costs of capacity expansion and facility operations. We represent uncertain parameters using a standard scenario tree (e.g., Ruszczyński and Shapiro 2003, pp 29-30). Given a finite number of scenarios and their probabilities, this problem can then be recast as a large-scale mixed-integer program, i.e., a "deterministic equivalent", that can be solved, in theory, by a commercial optimization code. As we shall see, however, only the smallest real-world instances may be tractable with this approach.

We overcome the intractability of the deterministic equivalent by applying dynamic column generation to a Dantzig-Wolfe reformulation of the problem (Dantzig and Wolfe 1960, Appelgren 1969). The Dantzig-Wolfe master problem represents a simplified deterministic equivalent for the problem, and subproblems generate columns for the master problem at each node of the scenario tree. The master problem exhibits structure that tends to yield integer solutions from its linear-programming (LP) relaxation, making it particularly easy to solve. When a facility can be expanded at most once over the planning horizon, model simplifications enhance performance. Specially

structured subproblems admit stronger formulations that further enhance performance, and "duals stabilization" for the master problem (e.g., du Merle et al. 1999) dramatically improves solution times for all problem variants.

The literature on stochastic capacity-planning problems is extensive: Luss (1982) and Van Mieghem (2003) present comprehensive surveys. Manne's seminal paper (Manne 1961), which models demand growth as an infinite-horizon stochastic process, stimulated much research on infinite-horizon models (e.g., Giglio 1970, Freidenfelds 1980). However, such models cannot incorporate the complex operational constraints that many real-world applications require.

More recent studies incorporate application-specific constraints. For instance, Sen et al. (1994) develop a two-stage model that integrates demand, capacity expansion, and budget constraints, although it assumes only continuous capacity-expansion decisions and a single capacity-expansion technology. The authors solve the model using a sampling-based stochastic-decomposition algorithm.

The assumptions of a discrete probability distribution for uncertain parameters and a finite planning horizon mean that a set of scenarios can represent uncertain outcomes resulting in a (possibly large-scale) mathematical programming problem. In this framework it is possible to include a detailed operational model and "strategic details" such as a variety of capacity-expansion technologies. Berman et al. (1994) present and solve a scenario-based multi-stage model with a single capacity-expansion technology. Chen et al. (2002) extend this concept to multiple capacity-expansion technologies, and also model economies of scale. However, both of these approaches assume continuous capacity-expansion decisions.

The use of integer variables to model fixed-charge cost functions and economies of scale adds considerable complexity. Eppen et al. (1989), Riis and Andersen (2002), Riis and Lodahl (2002), and Barahona et al. (2005) model these using integer variables in the first-stage of two-stage models.

In recent years, increased computing power and advances in optimization techniques have made it possible to develop and solve multi-stage stochastic integer-programming models. Ahmed et al. (2003) solve such problems with a special branch-and-bound procedure that incorporates

a heuristic upper-bounding method. Ahmed and Sahinidis (2003) and Huang and Ahmed (2005) propose approximation schemes that asymptotically converge to an optimal integer solution as the planning horizon lengthens.

Dynamic programming, though limited in its ability to integrate practical constraints, appears in a few recent applications. Laguna (1998) solves a two-stage model, which Riis and Anderson (2004) extend to multiple stages. Rajagopalan et al. (1998) present a multi-stage model with deterministic demand, but with uncertainty in the timing of the availability of new capacity-expansion technologies.

A multi-stage stochastic program with integer variables in all stages does not allow a nested Benders decomposition as does its continuous counterpart. In theory, LP-based branch and bound can solve the deterministic equivalent for such a problem, but practical instances usually exceed the ability of today's software and hardware to solve them. Decomposition procedures based on column generation are becoming more common, however, for solving large deterministic integer programs (e.g., Lübbecke and Desrosiers 2002). This has spawned new research in solving stochastic integer programs: Lulli and Sen (2004) use branch and price (column generation plus branch and bound) for stochastic batch-sizing problems; Shiina and Birge (2004) use column generation to solve a unit-commitment problem under demand uncertainty; Damodaran and Wilhelm (2004) model high-technology product upgrades under uncertain demand; and Silva and Wood (2004) solve a generic class of two-stage problems by branch and price.

We propose a new column-generation approach for solving multi-stage, stochastic, capacity-planning problems: our master problem and subproblems differ substantially from those developed by other researchers. Importantly, the generality of our approach should lend itself to applications in many industries.

Our research relates most closely to that of Ahmed et al. (2003). These authors present a multi-stage stochastic capacity-planning model that includes continuous as well as binary capacity-expansion decisions. They disaggregate the continuous variables using the reformulation strategy of Krarup and Bilde (1977), which enables a strong problem formulation. Our approach differs in

three major aspects:

1. We disaggregate the binary capacity-expansion decisions rather than continuous ones.

2. Random demand parameters directly determine a facility's capacity requirements in Ahmed et al. (2003), and operational constraints are simple: total installed capacity must meet or exceed demand (although, in theory, their model can accommodate more complicated operational constraints). Our approach incorporates a general operational-level submodel, which meets demand using installed capacity however the modeler deems fit.

3. Ahmed et al. (2003) solve their mixed-integer program using an LP-based branch-and-bound algorithm with a heuristic upper-bounding scheme; we use column generation.

The remainder of this paper develops as follows. The next section describes a general, multi-stage, stochastic, capacity-planning model with discrete capacity-expansion decisions. We formulate this problem as a deterministic-equivalent mixed-integer program. A revised reformulation, using the technique of "variable splitting", then enables a Dantzig-Wolfe decomposition whose master problem is likely to be stronger than that derived from the original formulation. Section 3 explores the strength of the decomposition. Section 4 formulates a restricted form of the general model which allows at most one expansion of each facility over the planning horizon. In section 5 we present computational results achieved by applying the general and restricted formulations to a capacity-planning problem for an electricity-distribution network. The final section presents conclusions.

2 A Multi-Stage, Stochastic, Capacity-Planning Model

We follow Ahmed et al. (2003) and represent uncertainty using a *scenario tree* $\mathcal{T}$ over T decision *stages*. For simplicity, we think of these stages occurring at evenly spaced increments of time. The uncertain parameters represent a discrete-time stochastic process defined on a finite probability space. The scenario tree at each stage t consists of a set of *nodes* that represents collections of states of the world that are indistinguishable up to time t. We denote by $n \in \mathcal{N}$ the set of nodes of the scenario tree.

Stage 1 comprises only $n = 1$, the root node of $\mathcal{T}$, which is where all scenarios have the same realization. For each node $n \in \mathcal{N}$, ϕ_n denotes the probability that the state of the world associated with node n occurs. $\mathcal{T}_n$ denotes the *successors* of n which we define to include n itself. Thus, $\mathcal{T}_n$ denotes n plus all nodes "below" n in the tree. $\mathcal{P}_n$ denotes the set of all *predecessors* of n, which we define to include n itself. Thus, $\mathcal{P}_n$ denotes n plus all nodes "above" n in the tree. For any leaf node n in the tree, $\mathcal{P}_n$ defines a *scenario*. We now present the compact formulation of our stochastic capacity-planning model.

Data

$\mathbf{c}_n$	discounted cost vector for expanding capacity of system facilities at scenario-tree node n
$\mathbf{q}_n$	discounted cost vector for operating the system at scenario-tree node n
$\mathbf{u}_0$	vector of initial capacities of facilities
V_n	matrix that converts operating decisions and/or activities into capacity utilization at scenario-tree node n
U_{hn}	non-negative matrix that converts capacity-expansion decisions at scenario-tree node h into available operating capacity at successor node $n \in \mathcal{T}_h$

Variables

Capacity-expansion decisions could be very complicated, because we might use various technologies to expand a facility f, and decisions in one time period could affect decisions in another. For simplicity, the model we describe here assumes that facility f can be expanded at scenario-tree node n or not, and can be expanded multiple times over the planning horizon. This gives:

$\mathbf{x}'_n$	vector of binary decisions for capacity expansion of facilities at scenario-tree node n. Specifically, $x'_{fn} = 1$ if facility f is expanded at node n, 0 otherwise.
$\mathbf{y}_n$	vector of continuous and/or discrete operating decisions at scenario-tree node n

Formulation

$$\textbf{CF}: \quad \min \quad \sum_{n\in\mathcal{N}} \phi_n \left(\mathbf{c}_n^\top \mathbf{x}'_n + \mathbf{q}_n^\top \mathbf{y}_n\right) \tag{1}$$

$$\text{s.t.} \quad V_n \mathbf{y}_n \leq \mathbf{u}_0 + \sum_{h\in\mathcal{P}_n} U_{hn} \mathbf{x}'_h \;\; \forall\, n \in \mathcal{N}, \tag{2}$$

$$\mathbf{y}_n \in \mathcal{Y}_n \;\; \forall\, n \in \mathcal{N}, \tag{3}$$

$$\mathbf{x}'_n \in \{0,1\} \;\; \forall\, n \in \mathcal{N}. \tag{4}$$

Random demands, costs etc., appear as the parameters subscripted by n in the model (excluding ϕ_n). Constraints (3) represent generic relationships between the operational variables $\mathbf{y}_n$, independent of all $\mathbf{x}'_h$. Constraints (2) ensure that adequate capacity exists to satisfy the operational requirements $V_n\mathbf{y}_n$ at node n. The matrices U_{hn} can model lags between when capacity-expansion decisions are executed and when capacity becomes available, and, more generally, can model capacity that increases or decreases over time after installation.

Constraints (2) and (3) can handle a general operational model at each node of the scenario tree. If a set of discrete capacity-expansion decisions adequately models continuous capacity expansions, the "(SCAP)" model of Ahmed et al. (2003) may be viewed as an instance of CF. In particular, this instance sets $\mathbf{q}_n = \mathbf{0}$ and defines constraints (3) as $\mathbf{y}_n = \mathbf{d}_n$, where $\mathbf{d}_n$ represents demands at node n.

Capacity-planning problems like CF typically have weak LP relaxations, and that makes them difficult to solve. The scale imposed by a scenario tree, especially when some components of $\mathbf{y}_n$ must be integer, exacerbates this difficulty. On the other hand, an optimization model over $\mathbf{y}_n \in \mathcal{Y}_n$, for a single node n, might be relatively easy to solve as a mixed-integer program. This structure suggests some form of decomposition.

2.1 A Split-variable Reformulation and Dantzig-Wolfe Decomposition

The classical approach to solving multi-stage stochastic linear programs uses nested Benders decomposition (e.g. Birge and Louveaux 1997, pp 234-236). In general, however, integer variables in

subproblems makes Benders decomposition inapplicable.

Our approach exploits Dantzig-Wolfe decomposition (Dantzig and Wolfe 1960) together with dynamic column generation (e.g., Lübbecke and Desrosiers 2002). As we shall later discuss, a straightforward Dantzig-Wolfe decomposition of CF could lead to a master problem that provides a weak lower bound on the optimal value of CF. Consequently, we first reformulate CF using a variable-splitting technique and then apply the decomposition. The split-variable formulation is

$$\textbf{SV}: \quad \min \quad \sum_{n\in\mathcal{N}} \phi_n \left(\mathbf{c}_n^\top \mathbf{x}'_n + \mathbf{q}_n^\top \mathbf{y}_n\right) \tag{5}$$

$$\text{s.t.} \quad \mathbf{x}_{hn} \le \mathbf{x}'_h \ \ \forall\, n \in \mathcal{N},\ h \in \mathcal{P}_n, \tag{6}$$

$$V_n \mathbf{y}_n \le \mathbf{u}_0 + \sum_{h\in\mathcal{P}_n} U_{hn}\mathbf{x}_{hn} \ \ \forall\, n \in \mathcal{N}, \tag{7}$$

$$\mathbf{y}_n \in \mathcal{Y}_n \ \ \forall\, n \in \mathcal{N}, \tag{8}$$

$$\mathbf{x}'_n \in \{0,1\} \ \ \forall\, n \in \mathcal{N}, \tag{9}$$

$$\mathbf{x}_{hn} \in \{0,1\} \ \ \forall\, n \in \mathcal{N},\ h \in \mathcal{P}_n. \tag{10}$$

The proof of the following proposition is obvious.

Proposition 1 *$(\mathbf{x}'_n, \mathbf{y}_n)_{n\in\mathcal{N}}$ is feasible for CF if and only if there exists $(\mathbf{x}_{hn})_{h\in\mathcal{P}_n, n\in\mathcal{N}}$ such that $(\mathbf{x}'_n, (\mathbf{x}_{hn})_{h\in\mathcal{P}_n}, \mathbf{y}_n)$ is feasible for SV. That is, CF and SV are equivalent.* ■

In SV, for each node n, and for each of its predecessor nodes $h \in \mathcal{P}_n$, we define new variables $\mathbf{x}_{hn}$ that indicate whether capacity expansions of facilities at scenario-tree node h contribute towards meeting the capacity requirement at node n. Here one may think of $\mathbf{x}_{hn}$ as *requests* for capacity expansions at nodes $h \in \mathcal{P}_n$ which, if granted, will jointly satisfy capacity requirements at node n. Constraints (7) accumulate such requests. The variables $\mathbf{x}'_n$ determine actual capacity expansions at node n and can be viewed as capacity *grants*. Thus the natural interpretation of constraints (6) is variables $\mathbf{x}_{hn}$ requesting capacity and variables $\mathbf{x}'_h$ granting capacity. (As an alternative, looking "down the tree" from node n, one may split $\mathbf{x}'_n$ into variables $\mathbf{x}_{nh}$, which indicate whether a capacity-expansion decision at node n is exploitable, non-exclusively, at successor node h; this equivalent

interpretation can be formalized by rewriting constraints (6) as $\mathbf{x}_{nh} \leq \mathbf{x}'_n \; \forall \, n \in \mathcal{N}, \; h \in \mathcal{T}_n$.)

The split-variable reformulation has some similarities to the reformulation that Krarup and Bilde (1977) use to strengthen lot-sizing models, and to the variable-disaggregation based reformulation used by Ahmed et al. (2003) for strengthening stochastic capacity-expansion models. Our model differs from those in that the split variables $\mathbf{x}_{hn}$ are binary and force binary capacity-expansion decisions $\mathbf{x}'_n$ that control the amount of capacity expansion. In contrast, Ahmed et al. (2003) disaggregate continuous variables that force continuous and binary capacity-expansion decisions. (We do not consider continuous capacity expansions.) Their model strengthens because demand explicitly provides a lower bound on the capacity requirement of a facility, and this allows the computation of tighter constraints.

Variable splitting is a common technique used in stochastic programming to enable the decomposition of certain models. The conventional application of this approach decomposes a model by scenarios. The decomposed model can then be solved by a variety of approaches such as Lagrangian relaxation (Schultz 2003), the branch-and-fix coordination scheme (Alonso-Ayuso et al. 2003), or branch and price (Lulli and Sen 2004). Applied to CF, for each node $n \in \mathcal{N}$, this approach would split variables $\mathbf{x}'_n$ and $\mathbf{y}_n$, into variables for the stage t associated with n and all scenarios s that are indistinguishable at n. Thus, the split variables here would be $\mathbf{x}'_{ts}$ and $\mathbf{y}_{ts}$. Because all split variables for a particular node n correspond to the same realization of the random parameters, their values must be equal: "non-anticipativity constraints" impose this condition (e.g., Birge and Louveaux 1997, p 25). The scenario-decomposition approach relaxes the non-anticipativity constraints to decompose the problem by scenario. The number of non-anticipativity constraints can be very large as they must be imposed on all variables at each non-leaf node. This complicates scenario-decomposition procedures.

In contrast to scenario decomposition, the master problem resulting from our decomposition of SV is simpler as it only involves non-anticipativity constraints on the variables $\mathbf{x}'_n$, and not on $\mathbf{x}_{hn}$ or $\mathbf{y}_n$. Moreover, this structure allows us to decompose the problem by scenario-tree node, which results in smaller, more manageable subproblems (as described below).

2.2 Dantzig-Wolfe Reformulation of SV

The capacity-expansion constraints (6) in SV link capacity expansions across successors of a scenario-tree node; these are "complicating constraints" to what are otherwise a set of simpler (sub)problems, one for each scenario-tree node n. (Subproblem n includes split variables $\mathbf{x}_{hn}$ indexed over $h \in \mathcal{P}_n$, but these variables are not linked across subproblems. Thus, they may be viewed as alternative capacity-expansion choices for subproblem n alone.) Thus, we can use decomposition to partition the constraints of the split-variable formulation into two sets: the set of linking (complicating) constraints (6), and the set of constraints specific to scenario-tree node n, for which we define

$$\mathcal{X}_n = \left\{ (\mathbf{x}_{hn})_{h \in \mathcal{P}_n} \mid V_n \mathbf{y}_n \leq \mathbf{u}_0 + \sum_{h \in \mathcal{P}_n} U_{hn} \mathbf{x}_{hn},\ \mathbf{x}_{hn} \in \{0,1\}\ \forall\, h \in \mathcal{P}_n,\ \mathbf{y}_n \in \mathcal{Y}_n \right\}. \tag{11}$$

In what follows, we find it convenient in some situations to replace the notation $(\mathbf{x}_{hn})_{h \in \mathcal{P}_n}$ with the more "vector-oriented" notation $(\mathbf{x}_{nn} \cdots \mathbf{x}_{1n}) \equiv (\mathbf{x}_{nn}\ \mathbf{x}_{p(n)n}\ \mathbf{x}_{p(p(n))n} \cdots \mathbf{x}_{1n})$, where $p(n)$ denotes the direct predecessor of node n.

Let $\mathcal{J}_n$ denote the index set for the finite set of vectors $\mathcal{X}_n$, whereby $\mathcal{X}_n = \left\{ (\widehat{\mathbf{x}}_{nn} \cdots \widehat{\mathbf{x}}_{1n})^j \mid j \in \mathcal{J}_n \right\}$. We can then express any element of $\mathcal{X}_n$ through

$$(\mathbf{x}_{nn} \cdots \mathbf{x}_{1n}) = \sum_{j \in \mathcal{J}_n} (\widehat{\mathbf{x}}_{nn} \cdots \widehat{\mathbf{x}}_{1n})^j w_n^j,\ \sum_{j \in \mathcal{J}_n} w_n^j = 1,\ w_n^j \in \{0,1\}\ \forall\, j \in \mathcal{J}_n.$$

Each vector $(\widehat{\mathbf{x}}_{nn} \cdots \widehat{\mathbf{x}}_{1n})^j$ represents a collection of capacity-expansion requests from nodes $h \in \mathcal{P}_n$; satisfying these requests will ensure feasible system operation at node n. Hence we refer to each collection of requests as a *feasible expansion plan* (FEP).

Without loss of generality, we may assume that each FEP has associated with it at least one optimal operational plan $\widehat{\mathbf{y}}_n^j$, i.e., $\mathcal{J}_n$ simultaneously indexes FEPs and operational plans at scenario-tree node n. Thus, we can attach the operational costs $\mathbf{q}^\top \widehat{\mathbf{y}}_n^j$ to the w_n^j, and substitute the expression above for $(\mathbf{x}_{nn} \cdots \mathbf{x}_{1n})$, to obtain the Dantzig-Wolfe reformulation of SV. (See Dantzig and Wolfe 1960 as the seminal reference for models with continuous variables, and see Appelgren 1969 for the extension to integer variables.) We denote this multi-scenario, column-oriented master problem as "MP".

For each scenario node n, MP contains a group of columns with index set $\mathcal{J}_n$. Each $j \in \mathcal{J}_n$ corresponds to an FEP. For simplicity, we assume that MP is always feasible, i.e., $\mathcal{J}_n \neq \emptyset$ for any n. The formulation for MP follows:

Sets and Indices

$j \in \mathcal{J}_n$ FEPs for scenario-tree node n

Data

$\widehat{\mathbf{x}}_{hn}^{j}$ binary vector representing capacity-expansion requests at node h that form part of FEP j for node n

$\widehat{\mathbf{y}}_{n}^{j}$ operational plan used with FEP j

Variables

$\mathbf{x}'_n$ binary decision vector for capacity expansion of facilities at scenario-tree node n

w_n^j 1 if FEP j is selected for scenario-tree node n, 0 otherwise

Formulation

$$\textbf{MP:} \quad \min \quad \sum_{n\in\mathcal{N}} \phi_n \mathbf{c}_n^\top \mathbf{x}'_n + \sum_{n\in\mathcal{N}} \sum_{j\in\mathcal{J}_n} \phi_n \mathbf{q}_n^\top \widehat{\mathbf{y}}_n^j w_n^j \qquad [\text{dual variables}] \qquad (12)$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}_n} \widehat{\mathbf{x}}_{hn}^{j} w_n^j \leq \mathbf{x}'_h \quad \forall\, n \in \mathcal{N},\ h \in \mathcal{P}_n, \qquad [\boldsymbol{\pi}_{hn}] \qquad (13)$$

$$\sum_{j\in\mathcal{J}_n} w_n^j = 1 \quad \forall\, n \in \mathcal{N}, \qquad [\mu_n] \qquad (14)$$

$$w_n^j \in \{0,1\} \quad \forall\, n \in \mathcal{N},\ j \in \mathcal{J}_n,$$

$$\mathbf{x}'_n \in \{0,1\} \quad \forall\, n \in \mathcal{N}.$$

Note that dual variables correspond to constraints in the LP relaxation of MP, which we denote as MP-LP. Optimal dual variables for restricted versions of MP-LP (and the other master problem in section 4) will be extracted for purposes of column generation.

MP's objective function (12) minimizes expected capacity-expansion costs plus expected operational costs. Constraints (13) ensure that no FEP is chosen for any node without sufficient capacity having been installed (granted). "Convexity constraints" (14) select exactly one FEP for each n.

Naturally, the cardinality of $\mathcal{J}_n$ in MP will be huge, so we solve MP using *dynamic column generation*. First, we create a restricted master problem (RMP) in which each set $\mathcal{J}_n$ represents a modest-sized subset of all the FEPs at scenario-tree node n. We solve the LP relaxation of RMP (RMP-LP), which replaces $w_n^j \in \{0,1\}$ and $\mathbf{x}'_n \in \{0,1\}$ by $w_n^j \geq 0$ and $\mathbf{x}'_n \geq 0$, respectively. (The convexity constraints imply satisfaction of $w_n^j \leq 1$ and $\mathbf{x}'_n \leq 1$.) Given a solution to RMP-LP, we extract dual variables, and attempt to generate new columns corresponding to FEPs with negative reduced costs, by solving optimization subproblems (e.g., Barnhart et al. 1998, Lübbecke and Desrosiers 2002).

The cycle of solving RMP-LP, extracting duals, and generating new columns repeats until no columns price favorably, i.e., no columns with negative reduced cost can be found and so we have solved MP-LP to optimality. If the optimal solution to MP-LP happens to be integer, then we have solved MP. If not, we may resort to a branch-and-price algorithm, which generates columns within a branch-and-bound procedure (Savelsbergh 1997), or settle for solving the RMP as an IP in the hope of getting a good integer solution.

A column j for node n in MP has the form $[\phi_n \mathbf{q}_n^\top \widehat{\mathbf{y}}_n^j, (\widehat{\mathbf{x}}_{nn} \cdots \widehat{\mathbf{x}}_{1n})^j, 1]^\top$, where $\mathbf{q}_n^\top \widehat{\mathbf{y}}_n^j$ is the cost of the associated operational plan $\widehat{\mathbf{y}}_n^j$, and $(\widehat{\mathbf{x}}_{nn} \cdots \widehat{\mathbf{x}}_{1n})^j$ is the corresponding FEP. Given the optimal duals, $\widehat{\boldsymbol{\pi}}_{hn}^\top$ and $\widehat{\mu}_n$ from RMP-LP, we can identify the column $j \in \mathcal{J}_n$ having the most favorable reduced cost by solving the subproblem

$$\mathbf{SP}(n): \quad \min \quad \phi_n \mathbf{q}_n^\top \mathbf{y}_n - \sum_{h \in \mathcal{P}_n} \widehat{\boldsymbol{\pi}}_{hn}^\top \mathbf{x}_{hn} - \widehat{\mu}_n \tag{15}$$

$$\text{s.t.} \quad V_n \mathbf{y}_n \leq \mathbf{u}_0 + \sum_{h \in \mathcal{P}_n} U_{hn} \mathbf{x}_{hn}, \tag{16}$$

$$\mathbf{y}_n \in \mathcal{Y}_n, \tag{17}$$

$$\mathbf{x}_{hn} \in \{0,1\} \ \ \forall\, h \in \mathcal{P}_n. \tag{18}$$

Any solution $((\mathbf{x}_{nn} \cdots \mathbf{x}_{1n}), \mathbf{y}_n)$ of SP(n) with a negative objective value lets us create a new column for RMP, i.e., add a new element to $\mathcal{J}_n$. If no such solution exists for any n, then we have solved MP-LP to optimality.

3 Strength of the Decomposition

Dantzig-Wolfe decomposition of a large LP replaces the direct solution of a large-scale problem with a sequence of solutions of smaller, easier-to-solve problems. This indirect approach helps when solving large MIPs too. Decomposition of a MIP may also improve solution efficiency by defining a master problem whose LP relaxation is stronger than the relaxation of the original MIP. The SV reformulation of CF makes this possible in our case.

Recall that Dantzig-Wolfe decomposition expresses feasible points for the LP relaxation of the master problem as convex combinations of extreme points of the convex hulls of the set of feasible solutions for the subproblems. If each subproblem is simply an LP, then the convex hull of the set of feasible solutions is identical to the LP feasible region, and optimal solutions of the LP relaxation of the master problem will have the same value as the LP relaxation of the original problem. On the other hand, if the convex hull of a subproblem's feasible solutions is smaller than its LP feasible region—for example when the subproblem is an IP whose LP relaxation does not have integer extreme points—then the resulting master problem can have a tighter relaxation than that of the original MIP (Barnhart et al. 1998).

In CF, we might consider applying a conventional Dantzig-Wolfe reformulation to the capacity-expansion constraints (2). This results in subproblems for each scenario-tree node n with operational constraints (3) only over $\mathcal{Y}_n$. Indeed, in the not-uncommon case in which the $\mathbf{y}_n$ are continuous variables, the subproblems for a decomposition of CF are LPs, and no strengthening is obtained.

On the other hand, in the Dantzig-Wolfe decomposition of SV, the subproblems SP(n) can be viewed as single-period, discrete, capacity-expansion problems, which can be shown to be NP-hard (by transformation to minimum-cover problems). Thus, they do not have LP relaxations with integer extreme points, and so our decomposition gives a master problem whose LP relaxation is stronger than that of the SV model. For example, in one of our test-problem instances the optimal objective value for the LP relaxation of SV equals 123,388; in comparison, the corresponding MP-LP has an optimal objective value of 960,881, a 779% improvement.

It is also remarkable that MP-LP almost always has an optimal integer solution. Because the constraint matrix for this problem has coefficients that are either 0 or 1, it is easy to see that fixing the w_n^j to binary values leads to binary solutions for $\mathbf{x}'_n$ even when these variables are allowed to be continuous. For each node n in the scenario tree, the submatrix corresponding to the variables w_n^j has a perfect-matrix structure (Padberg 1974). These perfect submatrices prevent fractional solutions from occurring within a single block of variables w_n^j, $j \in \mathcal{J}_n$, thus making it less likely for fractional solutions to occur in MP-LP. (See Ryan and Falkner 1988 for an account of this effect in set-partitioning problems.) On the other hand, the constraint matrix of MP-LP as a whole may not be perfect since it has constraints on $\mathbf{x}'_n$ that link its (perfect) submatrices. Consequently, the interaction between these submatrices can give rise to fractional solutions, although we find that these occur only rarely in practice. (Section 5 provides an example of a fractional optimal solution.)

4 At Most One Capacity Expansion of a Facility

The general model SV allows a facility's capacity to be expanded more than once over the planning horizon. However, in some industries, over reasonably long horizons, planning for multiple expansions makes little sense because associated fixed charges are large, or "setups" have highly undesirable side effects.

This section therefore studies a version of SV that restricts each facility to being expanded at most once over the planning horizon. We also assume that U_{hn} is deterministic and does not evolve with the scenario tree or change over time, i.e., $U_{hn} = U \ \forall\, n \in \mathcal{N},\ h \in \mathcal{P}_n$. With these changes, SV becomes:

$$\textbf{SV1}'\text{:} \quad \min \quad \sum_{n \in \mathcal{N}} \phi_n \left(\mathbf{c}_n^\top \mathbf{x}'_n + \mathbf{q}_n^\top \mathbf{y}_n \right) \tag{19}$$

$$\text{s.t.} \quad \mathbf{x}_{hn} \leq \mathbf{x}'_h \ \ \forall\, n \in \mathcal{N},\ h \in \mathcal{P}_n, \tag{20}$$

$$V_n \mathbf{y}_n \leq \mathbf{u}_0 + U \sum_{h \in \mathcal{P}_n} \mathbf{x}_{hn} \ \ \forall\, n \in \mathcal{N}, \tag{21}$$

$$\sum_{h \in \mathcal{P}_n} \mathbf{x}'_h \leq 1 \ \ \forall\, n \in \mathcal{N}, \tag{22}$$

$$\mathbf{y}_n \in \mathcal{Y}_n \ \ \forall\, n \in \mathcal{N}, \tag{23}$$

$$\mathbf{x}'_n \in \{0,1\} \ \ \forall\, n \in \mathcal{N}, \tag{24}$$

$$\mathbf{x}_{hn} \in \{0,1\} \ \ \forall\, n \in \mathcal{N},\ h \in \mathcal{P}_n. \tag{25}$$

The reader will note that constraints (22) for non-leaf nodes are redundant. However, we include all these constraints in RMP-LP because, for reasons we cannot explain, the Dantzig-Wolfe algorithm tends to solver faster that way. (Of course, we turn off the optimizer's presolver when solving these LP relaxations so that it does not eliminate the redundant constraints.)

It is convenient to transform SV1′ into an equivalent formulation with fewer variables:

$$\textbf{SV1}: \quad \min \quad \sum_{n\in\mathcal{N}} \phi_n \left(\mathbf{c}_n^\top \mathbf{x}'_n + \mathbf{q}_n^\top \mathbf{y}_n\right) \tag{26}$$

$$\text{s.t.} \quad \mathbf{x}_n \le \sum_{h\in\mathcal{P}_n} \mathbf{x}'_h \ \ \forall\, n \in \mathcal{N}, \tag{27}$$

$$V_n \mathbf{y}_n \le \mathbf{u}_0 + U\mathbf{x}_n \ \ \forall\, n \in \mathcal{N}, \tag{28}$$

$$\sum_{h\in\mathcal{P}_n} \mathbf{x}'_h \le \mathbf{1} \ \ \forall\, n \in \mathcal{N}, \tag{29}$$

$$\mathbf{y}_n \in \mathcal{Y}_n \ \ \forall\, n \in \mathcal{N}, \tag{30}$$

$$\mathbf{x}'_n \in \{0,1\} \ \ \forall\, n \in \mathcal{N}, \tag{31}$$

$$\mathbf{x}_n \in \{0,1\} \ \ \forall\, n \in \mathcal{N}. \tag{32}$$

SV1′ and SV1 are equivalent problems by virtue of the following proposition.

Proposition 2 *There exists* $(\mathbf{x}_{hn})_{h\in\mathcal{P}_n}$ *with* $(\mathbf{x}'_n, (\mathbf{x}_{hn})_{h\in\mathcal{P}_n}, \mathbf{y}_n)$ *being feasible for SV*1′ *if and only if there exists* $\mathbf{x}_n$ *such that* $(\mathbf{x}'_n, \mathbf{x}_n, \mathbf{y}_n)$ *is feasible for SV*1.

Proof. Suppose $(\mathbf{x}'_n, (\mathbf{x}_{hn})_{h\in\mathcal{P}_n}, \mathbf{y}_n)$ is feasible for SV1′. Let $\mathbf{x}_n = \sum_{h\in\mathcal{P}_n} \mathbf{x}_{hn}$. To show that $(\mathbf{x}'_n, \mathbf{x}_n, \mathbf{y}_n)$ is feasible for SV1, it suffices to check that constraints (27), (28) and (32) are satisfied. Constraints (20) imply (27), and constraints (21) give (28). Moreover, $\mathbf{x}_n$ is binary because of (20) and (22).

Conversely, if $(\mathbf{x}'_n, \mathbf{x}_n, \mathbf{y}_n)$ is feasible for SV1, then let $\mathbf{x}_{hn} = \mathbf{x}'_h$ for all $h \in \mathcal{P}_n$. All constraints of SV1$'$ hold trivially, except for (21). These constraints are satisfied because

$$V_n \mathbf{y}_n \leq \mathbf{u}_0 + U\mathbf{x}_n \leq \mathbf{u}_0 + U \sum_{h \in \mathcal{P}_n} \mathbf{x}'_h = \mathbf{u}_0 + U \sum_{h \in \mathcal{P}_n} \mathbf{x}_{hn}.$$

This completes the proof. ■

We can now formulate a Dantzig-Wolfe decomposition of SV1, analogous to that of section 2.2, by defining

$$\mathcal{X}_n = \{\mathbf{x}_n \mid V_n \mathbf{y}_n \leq \mathbf{u}_0 + U\mathbf{x}_n,\ \mathbf{x}_n \in \{0,1\},\ \mathbf{y}_n \in \mathcal{Y}_n\},$$

and by expressing $\mathbf{x}_n$ through $\widehat{\mathbf{x}}^j_n$, $j \in \mathcal{J}_n$, which denote the enumerated feasible solutions in $\mathcal{X}_n$:

$$\mathbf{x}_n = \sum_{j \in \mathcal{J}_n} \widehat{\mathbf{x}}^j_n w^j_n,\ \sum_{j \in \mathcal{J}_n} w^j_n = 1,\ w^j_n \in \{0,1\},\ \forall\, j \in \mathcal{J}_n.$$

This gives a simplified master problem

$$\textbf{MP1}: \quad \min \quad \sum_{n \in \mathcal{N}} \phi_n \mathbf{c}_n^\top \mathbf{x}'_n + \sum_{n \in \mathcal{N}} \sum_{j \in \mathcal{J}_n} \phi_n \mathbf{q}_n^\top \widehat{\mathbf{y}}^j_n w^j_n \qquad \text{[dual variables]} \tag{33}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{J}_n} \widehat{\mathbf{x}}^j_n w^j_n \leq \sum_{h \in \mathcal{P}_n} \mathbf{x}'_h \quad \forall\, n \in \mathcal{N}, \qquad [\boldsymbol{\pi}_n] \tag{34}$$

$$\sum_{h \in \mathcal{P}_n} \mathbf{x}'_h \leq \mathbf{1} \ \ \forall\, n \in \mathcal{N}, \tag{35}$$

$$\sum_{j \in \mathcal{J}_n} w^j_n = 1 \ \forall\, n \in \mathcal{N}, \qquad [\mu_n] \tag{36}$$

$$w^j_n \in \{0,1\} \ \forall\, n \in \mathcal{N},\ \ j \in \mathcal{J}_n,$$

$$\mathbf{x}'_n \in \{0,1\} \ \forall\, n \in \mathcal{N},$$

and a simpler subproblem

$$\textbf{SP1}(n): \quad \min \quad \phi_n \mathbf{q}_n^\top \mathbf{y}_n - \widehat{\boldsymbol{\pi}}_n^\top \mathbf{x}_n - \widehat{\mu}_n \tag{37}$$

$$\text{s.t.} \quad V_n \mathbf{y}_n \leq \mathbf{u}_0 + U\mathbf{x}_n, \tag{38}$$

$$\mathbf{y}_n \in \mathcal{Y}_n, \tag{39}$$

$$\mathbf{x}_n \in \{0,1\}. \tag{40}$$

Recall that SP(n) includes binary variables $\mathbf{x}_{hn}$ for all nodes $h \in \mathcal{P}_n$. In contrast, SP1(n) incorporates only binary variables $\mathbf{x}_n$. Thus, the number of binary variables in SP1(n) reduces by a factor of $|\mathcal{P}_n|$, which can make this subproblem easier to solve.

5 Computational Results

This section applies the SV and SV1 formulations to instances of a model for planning the capacity expansion of an electricity distribution network subject to uncertain demand. The details of this class of model have been described in Singh (2004), so we give only a brief description. A distribution network is the low-voltage part of the electricity system supplying customers from a single source (typically a substation connected to generating plants through the high voltage transmission system). For each demand realization, the distribution network of interest must operate in a radial (tree) configuration, which means that power flows from the source to each demand point along a unique path of power lines. Typically, each power line has a switch at either end that can be open or closed, and although the full network has an underlying mesh structure, it is operated in a radial configuration by opening and closing these switches.

The configuration of the switches is determined by binary variables within constraints $\mathbf{y}_n \in \mathcal{Y}_n$, which must be satisfied at each scenario-tree node n. This makes each subproblem (SP(n) or SP1(n)) a challenging mixed-integer program in its own right. A "super-network model" for any subproblem provides a stronger LP relaxation for that subproblem. This model replaces certain sets of nodes and edges with simpler constructs involving "super-nodes" and "super-edges" which reduces the number of binary variables, and exploits some problem-specific valid inequalities; see Singh et al. (2004) for details. We make use of this strengthened formulation in all of the tests reported here.

We report results for seven problem instances, which differ by the number of stages in a binary scenario tree (five problems) and the number of stages in a ternary scenario tree (two problems). All problem instances derive from data for an actual distribution network in Auckland, New Zealand. The network data comprise 152 nodes, most of which are demand points, and 182

edges. All problem instances have been designed so that an optimal solution always exists in which no edge is expanded more than once over the planning horizon. This allows us to apply both SV1 and SV formulations and make direct comparisons.

We have implemented and tested our algorithms on a desktop computer with a Pentium IV 2.6 GHz processor, and 1 GB of RAM. We generate all models, and implement our decomposition algorithms within the Mosel algebraic modeling system, version 1.24, from Dash Optimization. The LP restricted-master problems are solved with Xpress Optimizer, version 14.24, also from Dash Optimization, but the MIP subproblems and the deterministic-equivalent models are solved with CPLEX, version 9.0 from ILOG, Inc.

Solver settings remain constant throughout all tests. All MIPs are solved with default parameter settings except that Gomory cuts are turned off and a moderate level of probing is used (CPX_PARAM_PROBE = 2). All subproblems are solved to optimality and the deterministic-equivalent problems are solved with a relative optimality tolerance of 1.0%. The time to solve each problem instance is limited to 7,200 seconds.

Observe that any (nontrivial) instances of RMP-LP will be infeasible unless one feasible column (FEP) exists for each scenario-tree node. We could use the classical "Phase I" approach to finding an initial feasible solution (e.g., Dantzig and Thapa 2003, pp 291-292), but it is simpler to guarantee such a solution by seeding the master problem with one FEP for each scenario-tree node. Except for trivially infeasible problems, an FEP for each node that requires all possible capacity expansions will surely be feasible, so those generate our initial columns.

Any such FEP translates into a column in RMP-LP that has coefficients of 1 in the capacity-expansion constraints for each facility, a coefficient of 1 in the convexity constraint for the corresponding scenario-tree node, and 0s elsewhere. Note that our application imposes no operational costs, so these initial columns, as well as the columns generated later, all have cost coefficients of 0.

At each iteration of the Dantzig-Wolfe decomposition, a lower bound $\underline{z}_{LP}$ on z^*_{LP}, the optimal objective value for MP-LP, is readily available. In particular, using the arguments in Wolsey (1998,

p 189), it is easy to show that

$$\underline{z}_{LP} = z_{LP} + \sum_{n \in \mathcal{N}} \delta_n \leq z^*_{LP}, \tag{41}$$

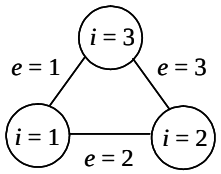
where z_{LP} and δ_n denote the optimal objective values for RMP-LP and SP(n) at the current iteration, respectively. Note that this lower bound is only valid when "full pricing" is invoked, i.e., after all subproblems SP(n), $n \in \mathcal{N}$ have been solved to optimality. At any particular iteration, it is easy to compute an upper bound $\overline{z}_{IP}$ on the optimal integer objective of MP by solving the integer RMP (RMP-IP) with the existing set of columns (assuming this is feasible). We define the (relative) optimality gap for the master problem, "MP-Gap", as $100\% \times (\overline{z}_{IP} - \underline{z}_{LP})/\underline{z}_{LP}$. MP-Gap gives an optimality check on our algorithm which can be used to terminate the Dantzig-Wolfe decomposition early if it has decreased to a tolerable level.

Observe that when the solution to RMP-LP is fractional, we must solve RMP-IP to obtain $\overline{z}_{IP}$, which can be expensive if carried out at every iteration. Thus for the overall efficiency of the algorithm, the number of such checks should be minimized. As an empirical rule, we start checking the MP-Gap at the first iteration when the gap between the RMP-LP objective and the lower bound, "LP-Gap", reaches 80% of a (preset) termination tolerance. For instance, for a termination tolerance of 5%, we start checking MP-Gap when LP-Gap reaches 4%. After the first check, we re-solve RMP-IP with a branch-and-bound algorithm only when RMP-LP yields fractional solutions for five consecutive iterations. We demonstrate the effect of termination tolerances on solution times later.

Unfortunately our Dantzig-Wolfe master problems suffer from severe dual degeneracy. Consequently, convergence using a conventional Dantzig-Wolfe algorithm is slow, ranging from hours to days. To improve convergence, we apply "duals stabilization" in the RMP-LP, and compare two different methods: du Merle et al. (1999) describe the first, which we call "du Merle stabilization"; the other simply generates interior-point dual solutions by solving RMP-LP using an interior-point algorithm. For lack of a better phrase, we call this technique "interior-point duals stabilization".

The optimal solutions of MP-LP are invariably integer in our test problems. Consequently, we have not required a full branch-and-price solution procedure. It is interesting to note, however,

that fractional optimal solutions are possible, at least in the master problem of the SV formulation; see figure 1 for an example network and figure 2 for the corresponding MP-LP.



Edge (e)	Initial Capacity (U_{e0})	Additional Capacity (U_e)
1	0	1
2	0	1
3	0	1

	Costs (C_{en})	
Edge (e)	n = 1	n = 2
1	1	0
2	1	0
3	0	0

Demand	Demand (D_{in})	
node (i)	n = 1	n = 2
1	0.75	1
2	0.75	1
3	-1.5	-2

Figure 1: Data for an example in which the master problem of SV formulation has a fractional LP solution. The diagram on the left represents a distribution network with three edges that connect supply node 3 to demand nodes 1 and 2. The tables on the right contain data for a single-scenario, 2-stage problem instance, i.e., $\mathcal{N} = \{1, 2\}$; here, $U_{ehn} = U_e$, $\forall\, n \in \mathcal{N}$, $h \in \mathcal{P}_n$.

		c_{11}	c_{21}	c_{31}	c_{12}	c_{22}	c_{32}							
		1	1	0	0	0	0							
		x'_{11}	x'_{21}	x'_{31}	x'_{12}	x'_{22}	x'_{32}	w_1^1	w_1^2	w_1^3	w_2^1	w_2^2		
		0.5	0.5	1	1	1	1	0	0.5	0.5	0.5	0.5		
	e													
n = 1, h = 1	1	1						-1	0	-1			≥	0
	2		1					-1	-1	0			≥	0
	3			1				-1	-1	-1			≥	0
n = 2, h = 1	1	1									-1	0	≥	0
	2		1								-1	0	≥	0
	3			1							-1	0	≥	0
n = 2, h = 2	1				1						-1	-1	≥	0
	2					1					-1	-1	≥	0
	3						1				-1	-1	≥	0
n = 1								1	1	1			=	1
n = 2											1	1	=	1

Figure 2: Constraint matrix and LP solution to the master problem of SV for the 2-stage single-scenario problem specified in figure 1. The solution is fractional.

The fractions arise from the interaction of requests by scenario-tree nodes 1 and 2 for capacity expansions on edges 1 and 2 at scenario-tree node 1. Interestingly, however, an alternate, integer, optimal solution exists: $x'_{11} = 0$, $x'_{21} = 1$, $x'_{31} = 1$, $x'_{12} = 1$, $x'_{22} = 1$, $x'_{32} = 1$, $w_1^1 = 0$, $w_1^2 = 1$, $w_1^3 = 0$, $w_2^1 = 0$ and $w_2^2 = 1$.

Scenario-tree Statistics			Deterministic Equivalent		Dantzig-Wolfe Decomposition			
Stages (number)	Scenarios (number)	Scenario-tree nodes (number)	SV-DE (CPU sec.)	SV1-DE (CPU sec.)	SV-DW-M (CPU sec.)	SV-DW-I (CPU sec.)	SV1-DW-M (CPU sec.)	SV1-DW-I (CPU sec.)
2	2	3	7.5	2.5	20.4	55.9	4.3	17.7
3	4	7	(1.8%)	1457.6	-	203.5	95.0	55.8
4	8	15	(42.2%)	(34.9%)	-	2852.3	638.1	284.5
5	16	31	(69.3%)	(65.6%)	-	(85.1%)	3624.2	1212.8
6	32	63	*	*	-	-	-	4301.4
5	81	121	*	*	-	(26.9%)	7043.5	2812.3
6	243	364	*	*	-	-	-	(7.6%)

Table 1: Solution times for each procedure. The values in parentheses are the relative optimality gaps achieved at 7,200 seconds. An asterisk denotes that no integer feasible solution was found in 7,200 seconds and a dash indicates that the optimality gap was more than 100%.

We use the following abbreviations to denote the various formulations discussed in earlier sections.

Abbreviation	**Formulation and Solution Procedure**
SV-DE	general split-variable formulation SV, solved as a deterministic equivalent
SV1-DE	specialized split-variable formulation SV1 that allows the expansion of a facility at most once in a scenario, solved as a deterministic equivalent
SV-DW-M	Dantzig-Wolfe decomposition of SV with du Merle duals stabilization
SV-DW-I	Dantzig-Wolfe decomposition of SV with interior-point duals stabilization
SV1-DW-M	Dantzig-Wolfe decomposition of SV1 with du Merle duals stabilization
SV1-DW-I	Dantzig-Wolfe decomposition of SV1 with interior-point duals stabilization

Table 1 displays the scenario-tree statistics for the seven problem instances, along with their solution times as deterministic equivalents, or using Dantzig-Wolfe decomposition. These results illustrate the power of decomposition in solving the larger problem instances.

The test problems are quite large. The largest SV instance we can solve with decomposition has 5 stages and 81 scenarios. It results in an SV-DE model having 158,602 binary variables and 194,864 constraints. The corresponding SV1-DE model has 81,070 binary variables and 117,332 con-

straints. Both models have 15,004 continuous variables. Neither CPLEX 9.0 nor Xpress Optimizer 14.24 can solve either of these problems in one day of computing time.

For this same instance, the largest subproblems for SV-DW have only 1,216 binary variables, while the SV1-DW subproblems have just 488 binary variables. The subproblems share the same 124 continuous variables and 800 constraints, and each solves in under 3 seconds on average. (Recall that the number of binary variables in the SV-DW subproblem for node n increases with its depth in the scenario tree. Thus, the subproblems for leaf nodes are the largest.)

The master problems for SV-DW and SV1-DW are of modest size, too. The restricted SV1-DW-I master problem for the 5-stage-81-scenario problem has only 23,161 variables in its last iteration, iteration 18 (see Table 3), and requires only 8.5 seconds to solve. In all iterations it has 44,165 constraints. The SV-DW master problem always has more constraints (see section 2.2), but its linear-programming relaxation usually solves quickly, too. The SV-DW master problem has 99,675 constraints for the 5-stage-81-scenario problem instance. Although SV-DW-I cannot solve this problem in under 7,200 seconds, at iteration 18 its LP master problem has 24,181 variables and solves in 7.3 seconds, while at iteration 92 the number of variables grows to 27,808, but still requires only 9.9 seconds to solve.

Our results show that interior-point duals stabilization is an important adjunct to the decomposition methodology, and that it is clearly superior to du Merle stabilization. For the 2-stage-2-scenario problem instance, the du Merle stabilization requires extensive tuning of its parameters to get SV-DW-M to converge. We also spent considerable effort tuning parameters for the 3-stage-4-scenario problem instance, but without success (as indicated by the dash). In contrast, the interior-point duals stabilization requires no tuning (other than ensuring that the standard "crossover" to a basic feasible solution is disabled), and it significantly outperforms the du Merle alternative. Nonetheless, the results of both duals-stabilization schemes exhibit the well-known tailing-off effect. Thus, terminating the Dantzig-Wolfe decomposition early by setting an acceptable optimality tolerance for MP-Gap may still give good solutions, without incurring the excessive computation time that it can take to reach optimality. Table 2 reports the time it takes SV-DW-I and SV1-DW-I to

satisfy tolerances of 5%, 1% and 0%.

Scenario-tree Statistics			Dantzig-Wolfe Decomposition					
Stages	Scenarios	Scenario-tree nodes	SV-DW-I (CPU sec.)			SV1-DW-I (CPU sec.)		
(number)	(number)	(number)	5%	1%	0%	5%	1%	0%
2	2	3	30.9	35.6	55.9	14.1	14.1	17.7
3	4	7	151.4	174.0	203.5	33.9	52.0	55.8
4	8	15	1886.3	2088.7	2852.3	188.1	188.1	284.5
5	16	31	16908.2	21355.2	24620.0	838.8	935.5	1212.8
6	32	63	-	-	-	2303.4	3286.7	4301.4
5	81	121	18005.3	21875.7	29820.7	1171.2	1370.4	2812.3
6	243	364	-	-	-	7407.1	11146.2	23637.9

Table 2: Computation times for SV-DW-I and SV1-DW-I to reach relative optimality gaps of 5%, 1% and 0%.

Table 3 reports the corresponding number of restricted master-problem iterations. As shown in this table, the SV-DW decomposition requires many more iterations to converge than SV1-DW. As observed above, the differences in the average solution times between the restricted master problems and subproblems for SV and SV1 are relatively small. So the large differences seen in overall solution times clearly result from SV-DW-I requiring many more iterations than SV1-DW-I. (It is interesting to see that the number of iterations for SV1-DW-I does not increase commensurately with problem size, at least for this application. This bodes well for solving even larger problems.)

It is important to note that the subproblems for this particular application are difficult, deterministic network-design problems (Johnson, Lenstra and Rinnooy Kan 1978). For this reason, and because we solve one subproblem for each scenario-tree node in each iteration, the total time spent solving subproblems is substantial. SV-DW-I spends 93.7% of its time solving subproblems while SV1-DW-I spends 98.2%, averaged over the problems both methods can solve. Clearly, then, any improvement in solution time for subproblems will improve overall solution time almost as much. All of the technology that has proved useful for solving deterministic network-design problems is worth evaluating for this purpose (e.g., Bienstock and Muratore 2000, Magnanti and Raghavan

Scenario-tree Statistics			Dantzig-Wolfe Decomposition					
Stages	Scenarios	Scenario-tree nodes	SV-DW-I (iterations)			SV1-DW-I (iterations)		
(number)	(number)	(number)	5%	1%	0%	5%	1%	0%
2	2	3	17	19	26	11	11	13
3	4	7	27	30	35	10	14	15
4	8	15	67	72	88	10	10	13
5	16	31	183	221	245	12	13	16
6	32	63	-	-	-	11	14	17
5	81	121	63	73	92	10	11	18
6	243	364	-	-	-	11	14	23

Table 3: Number of iterations for SV-DW-I and SV1-DW-I to reach relative optimality gaps of 5%, 1% and 0%.

2005).

As a final note, models that fit the SV or SV1 paradigm, but which have simpler subproblems, may solve very quickly. For instance, the multi-stage stochastic model of Riis and Anderson (2004) does fit the paradigm of SV, and its subproblems are simple knapsack problems, easily solved by dynamic programming.

6 Conclusions

We have described a general, compact formulation of a multi-stage stochastic integer-programming model for planning the capacity expansion of a production system with one or more production facilities. Capacity-expansions are discrete, and a scenario tree represents uncertainty.

We reformulate the compact formulation using a variable-splitting technique to give a general, split-variable model (SV) that allows multiple capacity expansions of a facility over the planning horizon. Based on SV we also devise a split-variable model (SV1) that restricts each facility to at most one capacity expansion over the planning horizon. A Dantzig-Wolfe reformulation of either model results in a master problem having a substantially stronger LP relaxation than the deterministic-equivalent formulation.

For each node n in the scenario tree, we define $\mathcal{P}_n$ to be the set of all predecessors of n, including n itself. Apart from variables $\mathbf{x}_{hn}$, which denote requests for capacity to be installed in nodes $h \in \mathcal{P}_n$, the variables in a subproblem SP(n) for the Dantzig-Wolfe reformulation of SV pertain only to node n. Indeed these variables can be viewed in the subproblem simply as alternative capacity-expansion options at node n of the scenario tree. As a result, the subproblems increase in difficulty only slightly with an increasing number of stages in a scenario tree. In SV1, the situation is even better, because the column-generation subproblems involve no variables (such as $\mathbf{x}_{hn}$) from predecessor nodes in the scenario tree. Thus, these subproblems do not become larger as the number of stages increases. This situation contrasts with scenario-decomposition methods in which the subproblems must cover an entire planning horizon, and so increase in size as more stages are added.

We have applied our methods to solve a capacity-planning problem for an electricity-distribution network, which requires the use of mixed-integer subproblems. However, the algorithm described is quite general. As long as good algorithms exist to solve them, the subproblems can incorporate arbitrary non-linearities or other complexities, which other applications may require.

The efficiency of column generation hinges on the use of a good duals-stabilization scheme for the master problem. For our application, the "interior-point duals stabilization" scheme, which obtains dual variables from an interior-point algorithm, greatly outperforms the well-known scheme of du Merle et al. (1999). Note that in our implementation of the interior-point method, we re-solve the master problems from a cold-start after adding a new set of columns. There is some potential to increase the speed of our algorithm by re-solving the master problems faster, using a suitable hot-start procedure for interior-point methods (e.g., Gondzio and Grothey 2003).

Our split-variable formulation uses inequality non-anticipativity constraints. The validity of these constraints relies on the assumption that capacity expansions are non-negative quantities. If this assumption were to be removed (for example, to admit facility closures) then the non-anticipativity constraints must be replaced by equalities in order to make SV correspond to CF. Based on this observation, it is tempting to suppose that general multi-stage stochastic integer-

programming problems might be profitably attacked using the approach outlined in this paper. Our experiments show that a formulation with an equality-constrained master problem can be solved using this approach, albeit with some increase in computational effort. For small problems, this is a modest increase, but the larger problems take up to ten times longer, so more research is necessary to make our approach viable for the general case.

Acknowledgments

The authors thank Vector Electricity for providing the data used in the computational experiments. Andy Philpott acknowledges support of the New Zealand Foundation of Research, Science and Technology under grant UOAX0203. Kevin Wood thanks the Office of Naval Research, the Air Force Office of Scientific Research, the Naval Postgraduate School, and the University of Auckland for their support. Kavinesh Singh acknowledges support of the New Zealand Tertiary Education Commission and Vector Electricity.

References

Ahmed, S., A. J. King, G. Parija. 2003. A multi-stage stochastic integer programming approach for capacity expansion under uncertainty. *Journal of Global Optimization* **26** 3–24.

Ahmed, S., N. V. Sahinidis. 2003. An approximation scheme for stochastic integer programs arising in capacity expansion. *Operations Research* **51** 461–471.

Alonso-Ayuso, A., L. F. Escudero, M. T. Ortuño. 2003. BFC, a branch-and-fix coordination algorithmic framework for solving some types of stochastic pure and mixed 0-1 programs. *European J. Oper. Res.* **151** 503–519.

Appelgren, L. H. 1969. Column generation algorithm for a ship scheduling problem. *Transportation Science* **3** 53–68.

Barahona, F., S. Bermon, O. Gunluk, S. Hood. 2005. Robust capacity planning in semiconductor manufacturing. *Naval Research Logistics* **52** 459–468.

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

Berman, O., Z. Ganz, J. M. Wagner. 1994. A stochastic optimization model for planning capacity expansion in a service industry under uncertain demand. *Naval Research Logistics* **41** 545–564.

Bienstock, D., G. Muratore. 2000. Strong inequalities for capacitated survivable network design problems. *Mathematical Programming* **89** 127–147.

Birge, J. R., F. Louveaux. 1997. *Introduction to Stochastic Programming*. Springer Series in Operations Research, Springer-Verlag, New York.

Chen, Z., S. Li, D. Tirupati. 2002. A scenario-based stochastic programming approach for technology and capacity planning. *Computers & Operations Research* **29** 781–806.

Damodaran, P., W. Wilhelm. 2004. Branch-and-price methods for prescribing profitable upgrades of high-technology products with stochastic demands. *Decision Sciences* **35** 55–81.

Dantzig, G. B., M. N. Thapa. 2003. *Linear Programming. 2*. Springer Series in Operations Research, Springer-Verlag, New York.

Dantzig, G. B., P. Wolfe. 1960. Decomposition principle for linear programs. *Operations Research* **8** 101–111.

du Merle, O., D. Villeneuve, J. Desrosiers, P. Hansen. 1999. Stabilized column generation. *Discrete Mathematics* **194** 229–237.

Eppen, G. D., R. K. Martin, L. Schrage. 1989. A scenario approach to capacity planning. *Operations Research* **37** 517–527.

Freidenfelds, J. 1980. Capacity expansion when demand is a birth-death random process. *Operations Research* **28** 712–721.

Giglio, R. J. 1970. Stochastic capacity models. *Management Science* **17** 174–184.

Gondzio, J., A. Grothey. 2002. Reoptimization with the primal-dual interior point method. *SIAM Journal on Optimization* **13** 842–864.

Huang, K., S. Ahmed. 2005. The value of multi-stage stochastic programming in capacity planning under uncertainty. In review, School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta.

Johnson, D. S., J. K. Lenstra, A. H. G. Rinnooy Kan. 1978. The complexity of the network design problem. *Networks* **8** 279–285.

Krarup, J., O. Bilde. 1977. Plant location, set covering and economic lot size: an $O(mn)$-algorithm for structured problems. *Numerische Methoden bei Optimierungsaufgaben, Band 3 (Tagung, Oberwolfach, 1976)*. Birkhäuser, Basel, 155–180. Internat. Ser. Numer. Math., Vol. 36.

Laguna, M. 1998. Applying robust optimization to capacity expansion of one location in telecommunications with demand uncertainty. *Management Science* **44** S101–S110.

Lübbecke, M. E., J. Desrosiers. 2002. Selected topics in column generation. To appear in Operations Research, Institut fur Mathematik, Technische Universitat Berlin, Germany, and HEC Montreal and GERAD, Montreal, Canada.

Lulli, G., S. Sen. 2004. A branch-and-price algorithm for multistage stochastic integer programming with application to stochastic batch-sizing problems. *Management Science* **50** 786–796.

Luss, H. 1982. Operations research and capacity expansion problems: A survey. *Operations Research* **30** 907–947.

Magnanti, T. L., S. Raghavan. 2005. Strong formulations for network design problems with connectivity requirements. *Networks* **45**(2) 61–79.

Magnanti, T. L., L. A. Wolsey. 1995. Optimal trees. *Network models*, *Handbook in Operations Research and Management Science*, vol. 7. North-Holland, Amsterdam, 503–615.

Manne, A. S. 1961. Capacity expansion and probabilistic growth. *Econometrica* **29** 632–649.

Manne, A. S. 1967. *Investments for Capacity Expansion: Size, Location and Time-Phasing*. Springer Series in Operations Research, MIT Press, Cambridge, Massachusetts.

Masse, P., R. Gibrat. 1957. Application of linear programming to investments in the electric power industry. *Management Science* **3** 149–166.

Padberg, M. W. 1974. Perfect zero-one matrices. *Mathematical Programming* **6** 180–196.

Rajagopalan, S., M. R. Singh, T. E. Morton. 1998. Capacity expansion and replacement in growing markets with uncertain technological breakthroughs. *Management Science* **44** 12–30.

Riis, M., K. A. Andersen. 2002. Capacitated network design with uncertain demand. *INFORMS Journal on Computing* **14** 247–260.

Riis, M., K. A. Andersen. 2004. Multiperiod capacity expansion of a telecommunications connection with uncertain demand. *Computers & Operations Research* **31** 1427–1436.

Riis, M., J. Lodahl. 2002. A bicriteria stochastic programming model for capacity expansion in telecommunications. *Mathematical Methods of Operations Research* **56** 83–100.

Ruszczyński, A., A. Shapiro. 2003. Stochastic programming models. *Stochastic programming*, *Handbook in Operations Research and Management Science*, vol. 10. Elsevier, Amsterdam, 1–64.

Ryan, D. M., J. C. Falkner. 1988. On the integer properties of scheduling set partitioning models. *European Journal of Operational Research* **35** 442–456.

Savelsbergh, M. 1997. A branch-and-price algorithm for the generalized assignment problem. *Operations Research* **45** 831–841.

Schultz, R. 2003. Stochastic programming with integer variables. *Mathematical Programming* **97** 285–309.

Sen, S., R. D. Doverspike, S. Cosares. 1994. Network planning with random demand. *Telecommunication Systems* **3** 11–30.

Shiina, T., J. R. Birge. 2004. Stochastic unit commitment problem. *International Transactions in Operational Research* **11** 19–32.

Silva, E. F., R. K. Wood. 2005. Solving a class of stochastic mixed-integer programs by branch-and-price. In review, Operations Research Department, Naval Postgraduate School, California.

Singh, K. J. 2004. Column-generation for capacity-expansion planning of electricity distribution networks. Andrew Mason, ed., *Proceedings of the 39th Annual OR Society of New Zealand Conference*. Operational Research Society of New Zealand, 76–85.

Singh, K. J., A. B. Philpott, R. K. Wood. 2004. Column-generation for survivable design of electricity distribution networks. In review, Department of Engineering Science, University of Auckland, New Zealand.

Van Mieghem, J. A. 2003. Capacity management, investment, and hedging: Review and recent developments. *Manufacturing & Service Operations Management* **5** 269–302.

Vanderbeck, F., L. A. Wolsey. 1996. An exact algorithm for IP column generation. *Operations Research Letters* **19** 151–159.