Imposing Connectivity Constraints in Forest Planning Models

Citation: Carvajal, Rodolfo, Miguel Constantino, Marcos Goycoolea, Juan Pablo Vielma, and Andres Weintraub. "Imposing Connectivity Constraints in Forest Planning Models." Operations Research 61, no. 4 (August 2013): 824–836.

As Published: http://dx.doi.org/10.1287/opre.2013.1183

Publisher: Institute for Operations Research and the Management Sciences (INFORMS)

Persistent URL: http://hdl.handle.net/1721.1/88144

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Version: April 9, 2013

Imposing Connectivity Constraints in Forest Planning Models

Rodolfo Carvajal
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332;
Escuela de Negocios, Universidad Adolfo Ibáñez, Santiago, Chile, rocarvaj@gatech.edu

Miguel Constantino
Centro de Investigação Operacional, Faculdade de Ciências da Universidade de Lisboa, miguel.constantino@fc.ul.pt

Marcos Goycoolea
Escuela de Negocios, Universidad Adolfo Ibáñez, Santiago, Chile, marcos.goycoolea@uai.cl

Juan Pablo Vielma
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, MA 02139;
Department of Industrial Engineering, University of Pittsburgh, Pittsburgh, PA 15260, jvielma@mit.edu

Andrés Weintraub
Departamento de Ingeniería Industrial, Universidad de Chile, Santiago, Chile; Instituto Milenio Sistemas Complejos de Ingeniería, Santiago, Chile, aweintraub@dii.uchile.cl

Connectivity requirements are a common component of forest planning models, with important examples arising in wildlife habitat protection. In harvest scheduling models, one way of addressing preservation concerns consists of requiring that large contiguous patches of mature forest are maintained. In the context of nature reserve design, it is common practice to select a connected region of forest, as a reserve, in such a way as to maximize the number of species and habitats protected. While a number of integer programming formulations have been proposed for these forest planning problems, most are impractical in that they fail to solve reasonably sized scheduling instances. We present a new integer programming methodology and test an implementation of it on five medium-sized forest instances publicly available in the FMOS repository. Our approach allows us to obtain near-optimal solutions for multiple time-period instances in fewer than four hours.

Key words: integer programming; cutting plane; natural resources

1. Introduction

During the last few decades, there has been much interest in incorporating environmental and aesthetic concerns into forest planning models. While there is no consensus on how these concerns should be fully addressed, a number of management practices have been consistently promoted by

voluntary initiatives and certifications such as the USA 2010-2014 Sustainable Forestry Initiative (SFI) (2010) or the Forest Stewardship Council (FSC) (2006). Three very important management practices with which we are concerned in this paper are as follows:

- Maximum clear-cut size constraints. Large clear-cut areas are unaesthetic, promote erosion and may have a negative impact on the wildlife living in surrounding areas. To counter this, maximum clear-cut size constraints dictate a maximum opening size of contiguous harvested areas. For example, the SFI recommends a maximum average area limit of 50 hectares. This norm is a legal requirement in many countries, for example Sweden (with a maximum of 20 ha.) and the United States (with maximums of 48 ha. in Oregon, and 101 ha. in Maine, to name a few).

- Conservation of Old-Growth-Forests. Wildlife protection has been mainly ensured by the existence of reserves. Nevertheless, the importance of managed forests as a complement to nature reserves has been widely recognized as a means to protect wildlife and biodiversity (Aldrich et al. 2004). While there is a variety of biodiverse habitats in forests, many species of animals concentrate in old-growth forest habitats, which are in short supply due to resource exploitation. Thus, it is common to require that large contiguous patches of mature forest (typically over 120 or 180 years old) be preserved. Rebain and McDill (2003a) review different sources on why small habitats may not accommodate wildlife species. Some reasons include insufficient territory for mating and breeding, lack of food, and increased predation and brood parasitism. Recent studies highlighting the global importance of old-growth forests as a carbon sink have sparked renewed interest in their conservation (Keith et al. 2009, Luyssaert et al. 2008). Protecting old-growth patches is also a way of mitigating the negative effects of maximum clear-cut size constraints (Barrett et al. 1998, Gustafson and Crow 1998).

- Natural reserve site selection. This consists of designing one or several contiguous sites for a natural reserve (wildlife refuge, national park, etc.) in such a way as to protect a specific list of species and/or preserve certain habitat types. Both the SFI and FSC standards are very specific in the need to have protected areas for threatened and endangered species. However, there is much debate on *how* reserves should be designed. For example, there are questions about the need for

wildlife corridors connecting different sites (Earn et al. 2000) and whether a few large sites are preferable to many smaller sites (Etienne and Heesterbeek 2000).

These three management practices have a very important feature in common: they all require that stands selected for harvesting or protection comply with some form of structural connectivity (or contiguity of stands). In this article, we study the problem of modeling structural connectivity (henceforth, connectivity, for short) in integer programming (IP) models. Our aim is not to advocate any particular way of addressing environmental concerns, but rather, to provide a more unified computational methodology capable of imposing connectivity in instances with over 1,000 stands and for different environmental constraints.

A number of authors have studied IP approaches for problems which involve the management practices mentioned above. The first attempts to incorporate maximum clear-cut size constraints are due to Thompson et al. (1973). Murray (1999) formalizes this problem in terms of basic stands or management units by introducing the Area Restriction Model (ARM). Goycoolea et al. (2005), McDill et al. (2002) and Constantino et al. (2008) propose different integer programming formulations for the ARM. Vielma et al. (2007) and Tóth (2005) introduce computational improvements for these formulations and Goycoolea et al. (2009) present a survey with computational results and modeling extensions. An important feature is that all of these models (with the exception of that found in Constantino et al. (2008)) address connectivity by explicitly enumerating all possible combinations of contiguous subsets satisfying certain properties. That these formulations work well in practice is due to the fact that maximum clear-cut size constraints are typically three to four times the size of an average stand in the instances considered. Hence, the number of potential clear-cut regions is not too large.

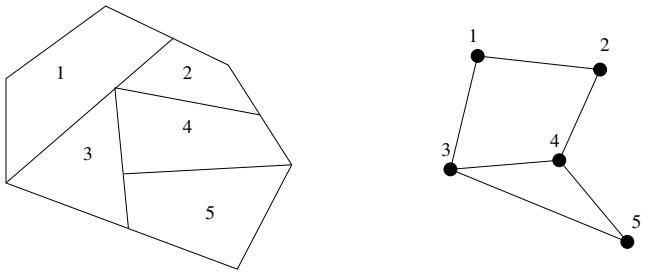
Caro et al. (2003) propose an extension of the ARM model that considers old-growth patches with enough area to be a wildlife habitat. Though they introduce an IP formulation, they use Tabu search to obtain feasible solutions for the model. Rebain and McDill (2003b) consider a similar model and propose solving it with integer programming. They use decision variables for every possible old-growth patch that has an area which exceeds a certain minimum requirement, and is

minimal in the sense that if a stand is removed from the patch, then it falls below the required area. This approach is illustrated with an instance containing 50 stands and three periods. Martins et al. (2005) use column generation to address the large number of variables in this formulation, solving atemporal (single-time period) instances with up to 400 stands.

The first attempts to quantitatively model the natural reserve selection problem date back to Kirpatric (1983). See Williams et al. (2005) for a comprehensive survey of articles that have followed. The problem of connecting dispersed reserves has also been considered. Sessions (1992) formulates this problem as a Steiner network problem and proposes a heuristic based on shortest paths. Williams (1998) uses an IP formulation with edge variables and flow conservation constraints to connect fixed reserves.

A number of articles study the problem of connectivity while also considering the shape of the natural reserve or old-growth patch. Tóth et al. (2006) use a two-objective approach to find a harvest schedule maximizing both the profit and the area of standing old-growth forest. Their model is based on the formulation of Rebain and McDill (2003b). This work is extended to minimize the perimeter of standing old-growth forest in Tóth and McDill (2008). They present results for a 50-stand data set. Öhman and Lämas (2005), and Öhman and Wikström (2008) also consider two-objective models. In the first work, the *Shape Index* (deviation from a circle) is heuristically minimized with simulated annealing, and in the latter, the perimeter is minimized as in Tóth and McDill (2008), but without requiring connectivity of old-growth forest. Öhman (2000) considers the creation of contiguous patches of old-growth forest with a large core area (area of a stand that is surrounded only by old-growth forest, wetlands, impediments, or lakes). Önal and Briers present a formulation for promoting compactness (2002) and for minimizing perimeter (2003). Contiguity is not guaranteed in these models. However, these same authors undertake the additional requirement of connectivity by using a characterization of trees in graphs in Önal and Briers (2006), performing computational experiments on a data set with 391 stands. Williams and ReVelle (1998) present an IP formulation that promotes, but does not guarantee, connectivity. Cerdeira et al. (2005) proposes a model that is specific to set covering models with connectivity constraints and applies it to a

Figure 1 A forest (left) modeled as an undirected graph G (right).



reserve selection problem with 496 stands. Dilkina and Gomes (2010) compare the effectiveness of solving compact formulations directly versus using cutting plane approaches (on edge-variables) for large formulations. Connectivity in reserve selection is also modeled using network flow techniques in Dilkina and Gomes (2010) and Shirabe (2005). As part of their work, Williams et al. (2005) provide a rich discussion on different types of connectivity and approaches used to model them.

Finally, we note that connectivity constraints are important for many other spatial optimization problems such as land acquisition (Williams 2002), political districting (Garfinkel and Nemhauser 1970) and facility location (ReVelle and Swain 1970).

Our Contribution

A forest can be represented by an undirected graph $G = (V, E)$, in which the vertices V correspond to stands, and the edges E correspond to pairs of adjacent stands (see Figure 1).

In this article, we describe a new integer programming framework for modeling connectivity in graphs that is well suited for the forestry applications described above because:

(a) It only uses vertex variables.

(b) It can model the requirement of one or multiple connected sub-graphs.

(c) It can model the containment of specific stands in the connected sub-graphs.

In particular, with respect to (a), we note that most existing forest planning models only use vertex-based (stand-based) decision variables. All of the integer programming formulations cited above, with the exception of Cerdeira et al. (2005) and Fügenschuh and Fügenschuh (2008), additionally use graph edge variables that result in significantly larger problems.

In addition to describing a model for imposing vertex (stand) connectivity, we describe a cutting-plane approach for implementing the model in practice, a heuristic for speeding up the cutting-plane algorithm, extensions for addressing different connectivity requirements, and techniques for strengthening the resulting formulations. To illustrate our approach, we focus on the specific problem in which we would like to solve the ARM subject to additional old-growth forest conservation constraints. However, it is easy to see that the approach extends to the other applications previously discussed. As we will see in the computational results section, our approach is able to handle instances with more than one thousand stands and multiple time periods. This is a considerable improvement on the results of Rebain and McDill (2003a) and Martins et al. (2005), who solve similar problems with 50 and 400 stands, 3 and 1 time periods, respectively, as well as on those observed in other forestry applications in which connectivity is considered. Moreover, we find that the formulations we propose enjoy very tight linear programming relaxation gaps in practice, and that for some types of connectivity, the corresponding models are easier to solve. This makes our proposed approach not only useful for forest planning applications, but also, as a tool by which to evaluate the performance of heuristics. All of our tests are performed on publicly available data sets.

2. An Example: Harvest scheduling model and old-growth forest

To motivate the need for connectivity constraints and to illustrate the applicability of our proposed approach, we consider a harvest scheduling problem with both maximum clear-cut constraints and old-growth conservation requirements.

The objective of this problem is to schedule the harvesting of forest stands to maximize profits, while preventing large clear-cut areas, providing a steady timber flow, maintaining a minimum average ending age of the forest and a connected (contiguous) region of old-growth forest (old-growth patch). With the exception of the connectivity requirements on the old-growth forest, all of these constraints can be modeled using IP in such a way that large instances can be solved (Goycoolea et al. 2009). We now describe one such model that utilizes a formulation of the clear-cut requirements that was introduced in McDill et al. (2002).

To describe the model, we use the notation and parameters given in Table 1.

V	The set of forest stands
$T = \{1, \ldots, N\}$	The planning horizon of time periods in which each stand can be harvested at most once
Y_t	Number of years in period t
$K_t = \sum_{s=1,\ldots,t} Y_s$	Number of years from the beginning of period one up to the end of period t
a_v	Area of stand v
α_{vt}	Volume of timber obtained, if stand v is harvested in period t
b_v	Age of trees in stand v, at the beginning of period 1
p_{vt}	The net present value of the profit of harvesting stand v in period t
H	Minimum average age of the forest at the end of the planning horizon
A_{max}	Maximum clear-cut area
Λ^+	Collection of contiguous groups of stands whose combined areas exceed A_{max} and are minimal for this property under inclusion. That is, a set of stands $\mathcal{C}$ is in Λ^+ if (a) The stands in $\mathcal{C}$ define a contiguous region of the forest, (b) the total area of the stands in $\mathcal{C}$ is strictly greater than A_{max} and (c) if we remove any stand from set $\mathcal{C}$, we obtain a set of stands not satisfying property (a) or (b).
A_{min}	Minimum area of the old-growth patch
O_{age}	Minimum age for a stand in the old-growth patch
L	Minimum fraction in which the volume of timber harvested in two consecutive periods can change
U	Maximum fraction in which the volume of timber harvested in two consecutive periods can change

Table 1 List of all the parameters of the model.

Note the following observations regarding Table 1:

- We consider two stands to be adjacent if they have a common edge boundary (reflecting the concept of structural connectivity or contiguity).
- We assume that the management actions (harvesting) occur at the end of a time period.
- The sets in Λ^+ are also called minimal infeasible clusters (Goycoolea et al. 2009) or paths (McDill et al. 2002).
- We assume $O_{age} > K_N$ and, hence, after a stand is harvested, it cannot belong to the old-growth patch in subsequent periods.

The formulation contains two sets of binary variables:

- y_{vt}: Harvesting variable. $y_{vt} = 1$ if stand v is scheduled for harvesting on period t and $y_{vt} = 0$

otherwise.

- z_{vt}: Patch variable. $z_{vt} = 1$ if stand v belongs to the old-growth patch in period t and $z_{vt} = 0$ otherwise.

and is given by

$$\max \quad \sum_{v \in V, t \in T} p_{vt} y_{vt} \tag{1a}$$

$$\text{s.t.} \quad z_{vq} + \sum_{t=1}^{q} y_{vt} \leq 1, \quad \forall v \in V, q \in T \tag{1b}$$

$$\sum_{v \in V} a_v z_{vt} \geq A_{min} \quad \forall t \in T \tag{1c}$$

$$z_{vt} = 0, \quad \forall v \text{ such that } b_v + K_t < O_{age}, \forall t \in T \tag{1d}$$

$$\sum_{v \in \mathcal{C}} y_{vt} \leq |\mathcal{C}| - 1, \qquad \forall \mathcal{C} \in \Lambda^+, \forall t \in T \tag{1e}$$

$$\sum_{v \in V} \alpha_{vt} y_{vt} \geq (1 - L) \sum_{v \in V} \alpha_{v,t-1} y_{v,t-1}, \quad \forall t \in T \setminus \{1\} \tag{1f}$$

$$\sum_{v \in V} \alpha_{vt} y_{vt} \leq (1 + U) \sum_{v \in V} \alpha_{v,t-1} y_{v,t-1}, \quad \forall t \in T \setminus \{1\} \tag{1g}$$

$$\sum_{v \in V} a_v \Big(b_v + K_N - \sum_{t \in T} (b_v + K_t) y_{vt} \Big) \geq \sum_{v \in V} a_v H \tag{1h}$$

$$y_{vt} \in \{0, 1\} \qquad \forall v \in V, \forall t \in T \tag{1i}$$

$$z_{vt} \in \{0, 1\} \qquad \forall v \in V, \forall t \in T \tag{1j}$$

$$\text{For each } t, \text{ the set of stands } Z_t = \{v : z_{vt} = 1\} \text{ is connected.} \tag{1k}$$

Objective (1a) maximizes the net present value of the schedule's profit. Constraints (1b) ensure that each stand unit is harvested at most once during the planning horizon and that a stand can be in the old-growth patch at the end of a period only if it has not been previously harvested. Constraints (1c) impose that the total area set aside as a part of the old-growth patch should exceed A_{min}. Constraints (1d) impose that in order for a stand to be considered an old-growth stand in time period t, its age in time t should be greater than or equal to O_{age}. Constraints (1e) are the so-called minimal infeasible or path constraints that enforce the maximum clear-cut requirements. In fact, constraints (1e) ensure that not all stands in a contiguous set $\mathcal{C}$ exceeding the maximum area constraint can be selected for harvesting. Constraints (1f) and (1g) are even volume

flow requirements, stating that the volume of timber harvested in period t cannot be a fraction U above or a fraction L below the volumes of timber in period $t-1$. Constraint (1h) requires that the average age of the forest at the end of the harvesting horizon is at least H years. Constraints (1i) and (1j) enforce integrality, and constraints (1k) enforce connectivity of the old-growth patch.

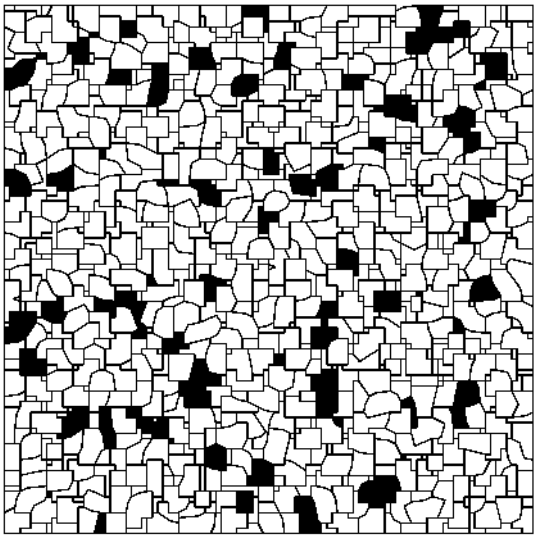
With the exception of (1k), all constraints of (1) are linear inequalities or integrality requirements. Hence, to transform model (1) to an IP formulation, we only need to replace (1k) by an appropriate IP formulation of connectivity. We will study such a formulation in Section 3. However, before that, we now illustrate why this connectivity constraint is needed for old-growth patches. To achieve this, we study the solutions of IP formulation (1a)–(1j) obtained by removing the connectivity requirements over the patch. We solve this formulation over three instances (*El Dorado*, *Shulkell*, and *FLG9A*) obtained from the repository of forestry instances of the Forest Management Optimization Site (FMOS) (2008). For all instances, we consider a three-period planning horizon. A detailed description of these instances is given in Section 4. For each instance, Table 2 presents the number of contiguous patches of old-growth forest (Patches) and the area of the largest of them (Largest), measured as a percentage of the minimum required area A_{min}, in the best feasible solution found. Figure 2 shows in black the stands selected to be part of the patch in the third period in a solution for the FLG9A forest.

Table 2 Number of connected patches obtained in FLG9A when not imposing connectivity. The entries in column "Largest Patch Size" are the size of the largest patch obtained, written as a percentage of the minimum required for old-growth forest.

	Number of Patches	Largest Patch Size (%)
El Dorado	70	17.6
FLG9A	57	5.8
Shulkell	10	25

We observe that without explicitly enforcing connectivity, we obtain solutions that are highly fragmented in terms of the selected old-growth forest. For example, as shown in Table 2, the old-growth stands selected in the FLG9A forest instance are divided into 57 disconnected pieces, each having an area of no more than 5.8% of the total requirement.

Figure 2 Old-growth area for FLG9A in time period 3 when not imposing connectivity. Stands in black are the ones selected for the old-growth patch. For simplicity, stands that are harvested in some period or are nonharvested, are shown in white.



3. Modeling Connectivity with Integer Programming

In this section, we study how to enforce connectivity constraints (1k). We begin by describing different types of connectivity requirements that may be considered in forest planning models. We then formally describe how these connectivity requirements can be enforced in integer programming models.

Formally, a patch is a set of contiguous stands in the forest sharing some uniform characteristic, (e.g., old growth stands). A set of contiguous stands, or a *connected* set of stands, is such that it is possible to travel between any two stands in the set through a path also fully contained in the set. In most forest planning models, decisions are made over time. In this context, if we are to impose connectivity, it is important to specify whether or not the patch (or, more precisely, the set of stands that form the patch) is allowed to change over time. If not, we say that the patch is *statically connected*. Otherwise, we say that it is *dynamically connected*. In the latter case, we assume that if a patch changes (e.g., stands are added, substracted or swapped) from one period to another, both the old and new patch must be connected. In some cases, it may be desirable to impose some limits on how much dynamically connected patches are allowed to change over time.

For example, it may be appropriate to require that from one time period to the next, the patch retain at least some specific number of stands. We call this a *temporal connectivity* requirement.

Rooted connectivity occurs when some predetermined set of stands is required to belong to a patch. For example, we may want an old-growth patch to contain some riparian zones. To make a distinction, we refer to *unrooted connectivity* if no specific stands are required to be in the patch.

It is often the case that we wish to impose some limit on the size of a patch. For example, we may want to impose a maximum area if the patch is to be a clear-cut zone, or a minimum area if the patch is to be an old-growth patch. We refer to these as *constrained connectivity* requirements.

In some applications with constrained connectivity constraints in which there is a total minimum area requirement, we may allow the existence of either a single large contiguous patch, or instead, several small patches, as long as each of these small patches meets some smaller minimum area condition. We refer to this type of flexible connectivity constraint (in which the model is allowed to decide if one or several patches will be defined) as a *multi-patch connectivity* requirement. Examples of this type of connectivity are found in Caro et al. (2003) and Rebain and McDill (2003a) in the context of old-growth patch modeling. We refer to the situation in which exactly one connected patch is required as *single-patch connectivity*.

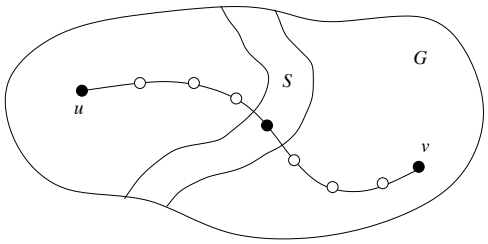
3.1. Background: Connectivity, Graphs and Integer Programming

In this section, we describe how each of the different connectivity requirements described in the previous section can be modeled with integer programming. We begin by introducing some notation. We next review the literature on graph connectivity and integer programming. Finally, we introduce new formulations for modeling connectivity.

Let $G = (V, E)$ be an undirected graph. Consider $U \subseteq V$. We define $G[U]$ to be the graph on U whose edges are precisely the edges of G having both endpoints in U. We say that G is *connected* if any two of its vertices are linked by a path in G. We say $U \subseteq V$ is connected if $G[U]$ is connected. A set $U \subseteq V$ is a *connected component* of G if it is maximally connected (i.e., $U \cup \{v\}$ is disconnected for all $v \in V \setminus U$).

There are a number of integer programming formulations for modeling connectivity in graphs. Most formulations use variables associated with the edges and are based on the Steiner-Tree Problem, and are concerned with static, unrooted, single-patch-connectivity. For a comprehensive survey of polyhedral and computational aspects of Steiner-Tree problems and their formulations, see Magnanti and Wolsey (1995). An example of such approaches applied to forest planning problems is Önal and Briers (2006). Formulations for the Node Weighted (or Prize Collecting) Steiner Tree Problem (Segev 1987) are similar, but consider variables for each node and each edge in the graph. Recent studies of Prize Collecting Steiner Tree Problem include Cordone and Trubian (2006), da Cunha et al. (2009), Ljubic et al. (2006), Lucena and Resende (2004). Maculan et al. (2003) and Shirabe (2005) propose compact formulations that use edge and vertex variables to impose connectivity through network flow conditions. Shirabe (2005) uses this type of formulation to solve a spatial unit allocation problem with 179 nodes. Dilkina and Gomes (2010) compare the use of compact network-flow based formulations to Steiner-Tree Problem (edge) formulations. Williams (2002) presents a compact formulation specialized for planar graphs, solving a 100-node instance. The use of *block* variables, (e.g., the use of one variable for each possible connected component) was considered by Martins et al. (2005) and Rebain and McDill (2003b) in the context of forest planning. However, the huge number variables is a serious handicap of such models. Using a column-generation approach, Martins et al. (2005) side-step this difficulty and manage to solve single-period instances with 400 stands. Cerdeira et al. (2005) exploit the special structure of connectivity-constrained set-covering problems to devise a specialized formulation for nature-reserve problems. Martin (1991) adapts a spanning tree characterization to impose connectivity that is equivalent to the Prize-Collecting Steiner Tree Formulation (Magnanti and Wolsey 1995). The paper of Fügenschuh and Fügenschuh (2008) is unique in proposing a connectivity formulation with variables associated with graph nodes and constraints associated with node-cut sets. Because the focus of this last paper is on sheet metal design, modelling connectivity is not explored in depth. Instead, the authors focus on addressing other difficult non-linear constraints, managing to solve instances with up to 30 nodes.

Figure 3 A uv-node cut S. Every path in G between u and v intersects S.



In what follows, we propose several new integer programming formulations for modeling node-connectivity. All of the models described require defining only a single variable for each node in the graph, and can easily be solved with a cut-generation approach. We find that defining variables for nodes, rather than for arcs, is most convenient for forest planning problems because the most important decision unit in such problems is the node (or stand). However, it should be noted that the approaches described next can also be used in the more general context of Node Weighted Steiner Tree Problems, as described above.

The integer programming techniques we describe are mainly based on the notion of node-cut sets, which we define next. Given nodes $u, v \in V$ that are non-adjacent ($\{uv\} \notin E$), a set of nodes $S \subseteq V \setminus \{u, v\}$ is a *node-cut set* separating u and v (or simply a *uv-node cut*) if there is no path between u and v in $G[V \setminus S]$. It is well known that given $U \subseteq V$ and a non-adjacent pair of nodes $u, v \in U$, then there exists a path in $G(U)$ between u and v if and only if all uv-node cuts S are such that $S \cap U \neq \emptyset$. Thus, connectivity of a graph can be characterized by its node-cut sets (see Figure 3).

Before describing the formulations, we present the following notation. We refer to Figure 4 to exemplify these definitions.

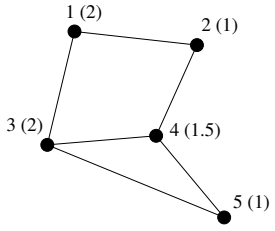
For $\{uv\} \notin E$ define

$$\Gamma(u, v) = \{S \subseteq V \setminus \{u, v\} : \ S \text{ is a minimal } uv\text{-node cut}\}.$$

In the graph in Figure 4, the set $\{3, 4\}$ belongs to $\Gamma(2, 5)$, but the set $\{1, 3, 4\}$ does not.

Consider $W > 0$. For each node $v \in V$, let $w_v \geq 0$ be an associated weight. Define, for each $v \in V$ the set

Figure 4 An example forest. The number in parenthesis, next to each vertex's label, represents a weight.



$$\mathcal{C}(v,W) = \{C \subseteq V : v \in C, \sum_{u \in C} w_u < W\}.$$

That is, $\mathcal{C}(v,W)$ corresponds to the set of all node-sets containing v but having total weight less than W. In Figure 4, the set $\{4,5\}$ belongs to $\mathcal{C}(5,3)$ (it has a total weight of 2.5), but the set $\{3,4,5\}$ does not (it has a weight of 4.5).

For each set of nodes $C \subset V$, define the neighborhood of C as follows,

$$\partial C = \{u \in V \setminus C : \exists v \in C, \{u,v\} \in E\}.$$

That is, ∂C corresponds to those vertices not in C that are adjacent to vertices in C. In Figure 4, for the set $C := \{4,5\}$ we have $\partial C = \{2,3\}$. Note that if $C \in \mathcal{C}(v,W)$, ∂C is a node-cut set separating v and each node not in $C \cup \partial C$.

Finally, for each node $v \in V$, define a 0-1 variable z_v indicating if we are to select node v or not. Let Z represent the set of selected nodes. That is,

$$Z = \{v \in V : z_v = 1\}.$$

In what follows, we are interested in systems of inequalities on the z variables that impose connectivity of set Z. In some cases, we consider the additional inequalities that are valid when, in addition to imposing that Z be connected, we impose the condition that Z meets a minimum weight requirement. That is,

$$\sum_{v \in V} w_v z_v \geq W. \tag{2}$$

- **Modeling connectivity**. To impose that Z be connected, the following inequalities suffice,

$$\sum_{w \in S} z_w \geq z_u + z_v - 1, \quad \forall S \in \Gamma(u,v), \forall u,v \in V, \{u,v\} \notin E. \tag{3}$$

Observe that if u and v are both selected ($z_u = z_v = 1$), the above constraints force a node of each minimal uv-node cut S to be selected as well.

For the case in which the graph G is disconnected, some inequalities (3) assume the form $z_u + z_v \leq 1$ when u and v belong to different connected components of G, since the empty set is a node cut separating u and v. If the graph G comprises several connected components, say $V_1, \ldots, V_q$, we may consider a stronger formulation (that is, one with a tighter linear programming relaxation bound) using the so-called *clique inequalities*

$$\sum_{p=1,\ldots,q} z_{u_p} \leq 1, \text{ for every choice of } u_1 \in V_1, \ldots, u_q \in V_q.$$

An equivalent way to strengthen the formulation is to consider additional binary variables $\eta_p = 1$ if the selected connected set is in component V_p and $\eta_p = 0$ otherwise, and the constraints

$$z_u \leq \eta_p, \qquad \forall u \in V_p,\ p = 1, \ldots, q \tag{4}$$

$$\sum_{p=1,\ldots,q} \eta_p \leq 1. \tag{5}$$

The formulation can be further strengthened (in terms of the linear programming relaxation bound) when constraints (2) are imposed in addition to the connectivity requirements. This can be done using *ring inequalities*, as described next. The main idea is as follows: any set of nodes C with weight less than W is too small to be a feasible set, so at least one of its neighbor nodes must be selected. Formally, given W and a vertex $v \in V$, the *unrooted ring inequalities* are as follows:

$$\sum_{u \in \partial C} z_u \geq z_v, \quad \forall C \in \mathcal{C}(v, W). \tag{6}$$

These inequalities were first proposed by Martins et al. (1999) in the context of multi-patch models (explained next).

Suppose now that a specific node $r \in V$ (e.g., a root) is required to belong to the set of nodes Z (rooted connectivity). The simplest way to model this requirement is by imposing $z_r = 1$. However, in this case, unrooted cut inequalities (3) can be replaced by the following set of inequalities.

$$\sum_{u \in S} z_u \geq z_v, \qquad \forall S \in \Gamma(r,v), \forall v \in V, \{r,v\} \notin E. \tag{7}$$

Observe that if v is selected ($z_v = 1$), the above constraints force a node of each minimal rv-node cut S to be selected as well. This is sufficient to impose that every pair of nodes u and v in Z are linked to each other in Z. Indeed, if there is a path between r and u and a path between r and v, then by joining the two paths we obtain a path between u and v. Rooted inequalities are stronger (in terms of the linear programming bound), as commonly noted in the literature on Steiner Tree Problems (Magnanti and Wolsey 1995).

If there is more than one root, we simply need to write constraints such as (7) for each pair $\{r,v\} \notin E$, where r is a root and v is not a root. If r_1, r_2 are two non adjacent roots, we can add the following constraint to strengthen the formulation.

$$\sum_{u \in S} z_u \geq 1, \qquad \forall S \in \Gamma(r_1, r_2). \tag{8}$$

This constaint enforces that the two roots must be connected. As in the unrooted case, if constraints (2) are imposed, the system can be further strengthened (in terms of the linear programming bound) with the following *rooted ring-inequalities*:

$$\sum_{w \in \partial C} z_w \geq 1, \quad \forall C \in \mathcal{C}(r, W). \tag{9}$$

Again, as in constraints (6), these constraints enforce that small disconnected sets are not allowed.

- **Multi-Patch connectivity**. In many applications, it is less relevant if Z is disconnected, as long as each connected component of Z meets a smaller minimum size or weight requirement. We call this weaker connectivity requirement a multi-patch connectivity requirement. Let W_{patch} be the minimum weight required by each connected component in Z. Note that this weight requirement is different from the weight requirement imposed by (2) in that the former is for each connected set in Z, as opposed to for all the nodes in Z. We can meet the multi-patch connectivity requirement in Z by *not* imposing constraints (3) and (8) and imposing the unrooted ring inequalities

$$\sum_{u \in \partial C} z_u \geq z_v, \quad \forall C \in \mathcal{C}(v, W_{patch}). \tag{10}$$

Suppose there are k roots $r_1, \ldots, r_k$ which must belong to the selected patches. In this case, we must proceed as in the unrooted case, defining constraints (10) for each node, and imposing that $z_r = 1$ for each root r. The presence of k roots does not guarantee the existence of k distinct patches. Rather, it guarantees that the optimal solution will contain an old growth area of at least A_{min}, that is divided into at most k distinct patches.

- **Dynamic connectivity.** So far we have only considered static patches. That is, we only consider patches that do not change with time. In many forest planning models, as in the one presented in Section 2, it may be possible, or even desired, to relax this condition. For example, in the context of forest harvest scheduling, it is possible to let younger old-growth-stands replace older old-growth-stands within old-growth patches as the forest ages. In order to model this, it is necessary to consider time-dependent variables z_{vt} for each $v \in V$ and $t \in T$, so that $z_{vt} = 1$ if and only if stand v is in the patch in time period t.

When using an unrooted model, use constraints

$$\sum_{w \in S} z_{wt} \geq z_{ut} + z_{vt} - 1, \quad \forall S \in \Gamma(u,v), \forall u, v \in V, \{u,v\} \notin E, \forall t \in T \tag{11}$$

For the case of a rooted model, use

$$\sum_{u \in S} z_{ut} \geq z_{vt}, \qquad \forall S \in \Gamma(r,v), \forall v \in V, \{r,v\} \notin E, t \in T \tag{12}$$

Constraints (11) and (12) impose exactly the same cut condition enforced by constraints (3) and (7), but do so in each time period.

In some situations, we may want to enforce *temporal connectivity*. Let us assume that, besides connectivity in each period, a minimum area of the patch, say A_{temp}, must be preserved in the patch from period t to period $t+1$. Connectivity in each period is expressed by constraints (11) and (12). Here, we describe two ways in which to enforce temporal connectivity. The first one uses additional binary variables, while the second one uses only variables z_{vt}, but requires exponentially many constraints.

The first way is as follows. Consider variables $\chi_{ut} = 1$ if the stand u is in the selected connected component, both in period t and $t+1$. Hence, we may consider constraints

$$\sum_{u \in V} a_u \chi_{ut} \geq A_{temp}, \quad \forall t \in T \setminus \{N\} \tag{13a}$$

$$\chi_{ut} \leq z_{ut}, \qquad \chi_{ut} \leq z_{u,t+1}, \quad \forall u \in V, \forall t \in T \setminus \{N\} \tag{13b}$$

Equations (13a) guarantee the required common area, while (13b) forces χ_{ut} to have the value zero if any of z_{ut} or $z_{u,t+1}$ is zero. Alternatively, the requirement of a common area A_{temp} can be expressed without additional variables χ, with the following set of constraints.

$$\sum_{u \in S} a_u z_{ut} + \sum_{u \in V-S} a_u z_{u,t+1} \geq A_{temp}, \qquad \forall S \subseteq V, \forall t \in T \setminus \{N\}. \tag{14}$$

To see this, suppose there is a period $t < N$ and a set $S^0 \subseteq V$ such that (14) does not hold. In this case there exists $\bar{z}$ such that $A_{temp} > \sum_{u \in S^0} a_u \bar{z}_{ut} + \sum_{u \in V-S^0} a_u \bar{z}_{u,t+1}$. However, since z is a binary variable, and since $a \geq 0$, we have that $\sum_{u \in S^0} a_u \bar{z}_{ut} + \sum_{u \in V-S^0} a_u \bar{z}_{u,t+1} \geq \sum_{u \in S^0} a_u \bar{z}_{ut} \bar{z}_{u,t+1} + \sum_{u \in V-S^0} a_u \bar{z}_{u,t+1} \bar{z}_{ut} = \sum_{u \in V} a_u \bar{z}_{ut} \bar{z}_{u,t+1}$. This means that the set of nodes u which belong to both connected components (e.g., those satisfying $\bar{z}_{ut} \bar{z}_{u,t+1} = 1$) have a total area of less than A_{temp}.

Conversely, suppose the common area in the connected sets $S^0 = \{u : z_{ut} = 1\}$ and $S^1 = \{u : z_{u,t+1} = 1\}$ is less than A_{temp}. Let $S = V - S^0$. Then $\sum_{u \in S} a_u z_{ut} + \sum_{u \in V-S} a_u z_{u,t+1} = \sum_{u \in S^0} a_u z_{u,t+1} = \sum_{u \in S^0 \cap S^1} a_u < A_{temp}$, so at least one of the constraints (14) is violated.

3.2. Solving the linear programming relaxations in a branch-and-cut algorithm

The formulations proposed here (either rooted or unrooted) encompass, in general, an exponential number of cut inequalities. Unlike in the ARM, these constraints cannot be explicitly enumerated and used directly in a formulation given to an integer programming solver. However, we can add them through a constraint generation procedure. From basic linear programming theory, it is known that only a few constraints are active in a basic solution. Hence, we need, at least in theory, to add only a reduced number of constraints to solve the linear programming relaxations.

We describe next a cutting plane algorithm to solve the linear programming relaxation of an integer programming model M that includes cut inequalities (3).

A *cutting plane algorithm* works as follows (Nemhauser and Wolsey 1999). Consider an initial linear programming formulation obtained by removing cut inequalities from the LP relaxation of M, or by keeping only a few of them. Let z^* be the corresponding optimal LP solution. Now solve the so-called *separation problem*, that is "check if all constraints (3) are satisfied by z^*, and if this is not the case, find a violated constraint (e.g., one not currently in the LP)". If all constraints not in the LP are satisfied by z^*, then it is not necessary to add any of them, and z^* is the optimal solution of the LP relaxation of M. Otherwise, add the violated constraint to the formulation and solve the new LP problem. Repeat the procedure until all constraints are satisfied. Since the number of constraints is finite, this procedure is finite, as long as the separation problems are solved exactly.

The good news is that the separation problem can be solved efficiently in this case, resulting in a practical constraint generation procedure. Given a solution z^* of the linear programming relaxation, the separation problem can be stated as an optimization subproblem: find nodes u and v such that $\{u, v\} \notin E$, and a set $S^* \in \Gamma(u, v)$ such that the sum $\sum_{w \in S^*} z^*_w$ is minimum among all sets $S \in \Gamma(u, v)$. If $\sum_{w \in S^*} z^*_w < z^*_v + z^*_u - 1$, the inequality (3) induced by u, v and S^* is violated; otherwise, if $\sum_{w \in S^*} z^*_w \geq z^*_v + z^*_u - 1$ for every pair of non adjacent nodes u and v, then the LP solution z^* is optimal.

In order to find a minimum node cut separating u and v, we may use a classical max-flow min-cut theorem (see, e.g., Nemhauser and Wolsey 1999): Given a graph with node capacities, the maximum flow between two non-adjacent nodes u and v equals the capacity of the minimum capacity node cut separating u and v. In our case, the node capacities are the values of the linear programming variables z^*_w. In practice, we transform the graph in such a way that each node is replaced by two opposite arcs, and use an efficient max flow algorithm to determine a minimum cut on the arcs that is then translated into a minimum cut in the nodes.

We repeat this cutting plane procedure to solve the linear programming relaxations at every node of the branch-and-bound procedure when solving the integer programming model M. We

note that in practice it is not necessary to complete the cutting plane procedure at each node of the branch-and-bound tree. That is, we may stop the procedure early, allowing some constraints (3) to be violated. This will provide an upper bound for the LP relaxation with all cut inequalities (the solution obtained from this early termination is optimal for a maximization problem with fewer cut constraints). If this upper bound is accurate, it might be enough to fathom the corresponding branch-and-bound node. If not, we branch and continue the cutting plane procedure on the child nodes. Most practical branch-and-cut algorithms carefully monitor the upper bounds provided by the cutting plane procedure and dynamically decide if it should be terminated. A simple version of this monitoring that is used in our implementation is described in Section 4.1.

The algorithm for solving the linear programming relaxation that considers inequalities (7) is analogous.

4. An Example

In this section, we describe a test of our proposed branch-and-cut algorithm on the harvest scheduling model given in Section 2. Recall from the discussion in Section 3 that the connectivity requirement (1k) in this problem is that the old-growth area should be constituted of a single unrooted dynamic patch. In order to compare the difficulty in solving this problem under different connectivity requirements, we also consider rooted and static variants of problem (1). The formulation of the static version of the problem is obtained by dropping the t index from the z_{vt} variables and replacing constraints (1b)–(1d) by

$$z_v + \sum_{t \in T} y_{vt} \leq 1, \quad \forall v \in V \tag{15a}$$

$$\sum_{v \in V} a_v z_v \geq A_{min} \tag{15b}$$

$$z_v = 0, \quad \forall v \text{ such that } b_v < O_{age} \tag{15c}$$

With these two modifications, we obtain the following four variants:

1. Rooted-Static: (1) with z_v instead of z_{vt} and (1k) replaced by (7).
2. Rooted-Dynamic: (1) with (1k) replaced by (12).

3. Unrooted-Static: (1) with z_v instead of z_{vt} and (1k) replaced by (3).
4. Unrooted-Dynamic: (1) with (1k) replaced by (11).

For the rooted instances, in each forest we consider three randomly selected roots that satisfy the minimum age requirements.

4.1. Implementation Details

We implement our branch-and-cut algorithm in C++ using the CPLEX 11.0 callable library (CPLEX 2007). For simplicity, we describe the details of this cutting plane algorithm only for the unrooted static version of the problem. The details for other versions are analogous. Let z^* denote the vector of values of the old-growth variables in the linear programming relaxation solution. For every pair of variables z_u, z_v such that $z^*_u + z^*_v > 1$, we try running our separation algorithm, obtaining either one or two minimum uv-node cuts. If the corresponding inequalities were violated by z^*, they are added to the formulation. In order to solve the separation problem, we use the implementation of the push-relabel algorithm available in EGlib (Espinoza and Goycoolea 2003). In some situations, we are able to find many different optimal cuts. In this case, we select two cuts: the cuts selected are the ones "closest" to nodes u and v, respectively. Whenever a minimum uv node cut is obtained, we check whether the corresponding inequality could be strengthened to a ring inequality, as mentioned in Section 3. That is, if the cut S is such that $S \in \mathcal{C}(v, A_{min})$ for some $w \in V$, then we add the corresponding unrooted ring inequality (6) instead of constraint (3).

At each iteration, before running the exact separation algorithm described above, we run a quick separation heuristic for unrooted ring inequalities that works as follows. First, we identify all of the connected components of the subgraph induced by $U = \{i \in V : z^*_i > 0\}$ using a simple depth-first-search algorithm (Cormen et al. 2001). Second, we identify all connected components S of $G[U]$ for which the total area is less than A_{min}. For each of these small components, we add the unrooted ring inequality (6) corresponding to the S cutset in the LP. Third, for the remaining connected components S of $G[U]$ (e.g., those with total area larger than or equal to A_{min}), we add a cut of type (3).

In preliminary tests, we discovered that the effectiveness of connectivity cuts (in terms of reducing the bound) decreases over time. This is known as a *tail-off* effect. To counteract this, we carry out a selective separation algorithm, only adding cuts at node zero (e.g., the root node of the branch and bound tree) when the value of the LP relaxation changes by a prescribed amount in the previous iteration. If the value of the LP relaxation does not change by more than 0.5% during 10 consecutive rounds of separation, we stop adding cuts. Whenever an integer solution is found in the branch-and-bound tree, we check if it complies with the connectivity requirements, in which case we find a feasible solution for the problem; otherwise, we add the violated constraints that are found to the problem.

Preliminary tests also show that CPLEX has trouble finding feasible solutions for some classes of problems. For this reason, we run the problem variants in an order such that feasible solutions for previous runs could be used as heuristic solutions for subsequent runs. Specifically, we run the variants in the order: 1) static-rooted 2) static-unrooted 3) dynamic-rooted 4) dynamic-unrooted. Then, we provide the best solution found in step 1) as a heuristic solution to steps 2) and 3) and the best solution between steps 2) and 3) as a heuristic to step 4).

4.2. Description of Forests

We consider six instances in this study. The first one is the hypothetical forest considered in Rebain and McDill (2003b) which we denote by *Rebain-McDill*. The others are five instances obtained from the repository of forestry instances of the Forest Management Optimization Site (FMOS) (2008): Gavin, Hardwicke, FLG9A, Shulkell and El Dorado. Tables 3 and 4 describe these instances and the parameters we use to define formulation (1) and its variants. For all instances, we consider a three time-period planning horizon, and we impose that the connectivity requirement should be met with a single patch.

Table 3 Characteristics of the FMOS instances used in our computational study and parameters used in the optimization model.

Name	Stands	Area (ha)	A_{max} (ha)
Rebain-McDill	50	1000	40
Gavin	352	6310	40
Hardwicke	423	6948	40
FLG9A	850	9999	48.6
Shulkell	1039	4498.7	16
El Dorado	1363	21147	48.5

Table 4 Parameters used in our computational study for solving the different forest planning problems.

Name	Value
L	0.15
U	0.15
H	40 Years
O_{age}	60 Years
A_{min}	20% of total area

4.3. Computational Results

The main goal of our computational tests is to assess the ability of the proposed branch-and-cut approach to obtain good solutions within a reasonable amount of time. The secondary goal is to assess the quality of the LP relaxation bound obtained when using the proposed formulations. In order to meet these goals, we run our implementation of the algorithm on each of the instances described in Section 4.2 on a Xeon Quad-Core server with 32 GB of RAM. We let each problem run for four hours. During each of the runs, we record the following values:

- t_{LP}: The time it takes to solve the LP relaxation of the problem at Node 0 (the root node of the branch-and-bound tree),
- U_{LP}: The objective function value obtained after solving the LP relaxation of the problem at Node 0 (the root node of the branch-and-bound tree),
- t_{IP}: The time (in seconds) it takes to solve the problem to optimality. For cases in which the problem is not solved in four hours, we let $t_{IP} = 14400$, corresponding to the time limit.
- L_{IP}: The objective function value of the best integer feasible solution obtained during the run,
- U_{IP}: For cases in which the problem is not solved to optimality in four hours, we let U_{IP} be the

best upper bound obtained during the run. For cases in which the problem is solved to optimality, we let $U_{IP} = L_{IP}$.

Note that U_{LP} and U_{IP} define upper bounds on the optimal objective function value and L_{IP} defines a lower bound. Define IP gap $= 100 \times (U_{IP} - L_{IP}) \div L_{IP}$ and LP gap $= 100 \times (U_{IP} - U_{LP}) \div U_{LP}$. We consider an integer feasible solution to be optimal if the associated IP gap is below 0.01%.

In Table 5, we present the results recorded for each data set and problem formulation. For those instances that are solved to optimality in fewer than four hours, instead of reporting the IP gap, we report the t_{IP} using square brackets.

Table 5 Summary of computational results obtained from solving 48 different forest planning problems. The entries in column "Root" can assume the value "R1", "R2", "R3" or "Unrooted". Value "R1" indicates that we have used the first of three randomly chosen roots. Values "R2" and "R3" are analogous. Value "Unrooted" indicates that we solve the unrooted model.

		Static			Dynamic		
Instance	Root	IP gap	LP gap	t_{LP}	IP gap	LP gap	t_{LP}
Rebain-McDill	R1	[0.58 s]	0.42%	0.09	[2.93 s]	0.77%	0.71
	R2	[1.93 s]	0.65%	0.09	[136.75 s]	1.87%	0.50
	R3	[0.12 s]	0.37%	0.06	[9.94 s]	0.79%	0.70
	Unrooted	[11.46 s]	0.49%	0.62	[6746.89s]	2.93%	0.70
Gavin	R1	0.01%	0.28%	5.03	7.29%	7.45%	17.59
	R2	0.09%	0.38%	4.25	6.42%	6.79%	12.43
	R3	0.02%	0.33%	7.08	5.47%	5.93%	16.57
	Unrooted	0.61%	0.84%	16.85	6.68%	7.05%	39.80
Hardwicke	R1	0.15%	0.23%	5.10	3.85%	4.06%	16.67
	R2	0.26%	0.38%	6.43	2.75%	2.88%	30.73
	R3	0.23%	0.39%	6.52	1.64%	1.75%	24.42
	Unrooted	0.41%	0.51%	16.52	2.32%	2.55%	49.64
FLG9A	R1	0.46%	2.20%	305.57	5.93%	6.03%	1219.25
	R2	0.12%	2.14%	161.67	5.94%	6.12%	780.52
	R3	0.05%	2.05%	137.51	5.91%	6.02%	1143.86
	Unrooted	3.19%	3.85%	393.22	7.62%	8.36%	591.41
Shulkell	R1	0.06%	0.19%	18.30	0.10%	0.23%	76.30
	R2	0.06%	0.18%	24.14	0.08%	0.22%	18.12
	R3	0.05%	0.17%	17.14	0.08%	0.23%	18.64
	Unrooted	0.05%	0.27%	81.65	0.55%	0.63%	175.09
El Dorado	R1	0.05%	0.08%	78.60	0.08%	0.08%	66.69
	R2	0.07%	0.10%	169.92	0.12%	0.14%	78.40
	R3	0.07%	0.11%	33.53	0.11%	0.11%	89.85
	Unrooted	0.14%	0.14%	502.76	0.14%	0.14%	1174.93

From this table, we fist observe that using our proposed formulations, we are able to obtain near-optimal ($< 1\%$ GAP) solutions for most instances and variants in the allotted time (35 out of 48 instances). Furthermore, even though some of the instances are relatively large (up to 1363 stands) with regard to similar problems solved in the literature, we can find good solutions for all the instances and variants (worst IP gap 7.62%, average 1.4%). This illustrates the effectiveness of our proposed branch-and-cut algorithm. As we mentioned before, this indicates a considerable improvement in terms of the size of the instances solved, over previous results in the literature and other forestry applications that involve connectivity.

A second observation is that the node zero LP relaxation values are usually very tight (the worst LP gap obtained is 8.4%, while the average LP gap is 1.9%) and can be obtained quite fast (in the worst case, the time to obtain the LP value is 21 minutes, while on average it requires 3 minutes). Finally, we see that the difficulty of solving these problems varies greatly when considering different forms of connectivity (average rooted-static IP gap 0.1%, average rooted-dynamic 0.7%, average unrooted-static 2.5%, and average unrooted-dynamic 2.9%). It is clear from our results that the proposed approach performs better on rooted problems than on unrooted ones. Analogously, the proposed approach performs better on static problems than on dynamic ones. With respect to forest data sets used, we observe that the difficulty is not always due to size as the hardest instances (FLG9A and Gavin) are not the largest.

In Figure 5, we graphically depict the optimal solutions of FLG9A (R1) and FLG9A (R2). As can be observed, the proposed formulation yields a solution in which the old-growth patch is connected. Thus, it can be seen that our formulation succesfully mitigates the effect observed in Figure 2, in which the solve the problem with the same data but without connectivity constraints. Visual inspection reaffirms the preliminary conclusions of Rebain and McDill (2003a) obtained on 50 stand and single period instances, namely, that promoting connectivity alone leads to old-growth patches that may have long and narrow shapes. Indeed, there is no mechanism in our models that promotes large interior area or low perimeter-to-area ratios.

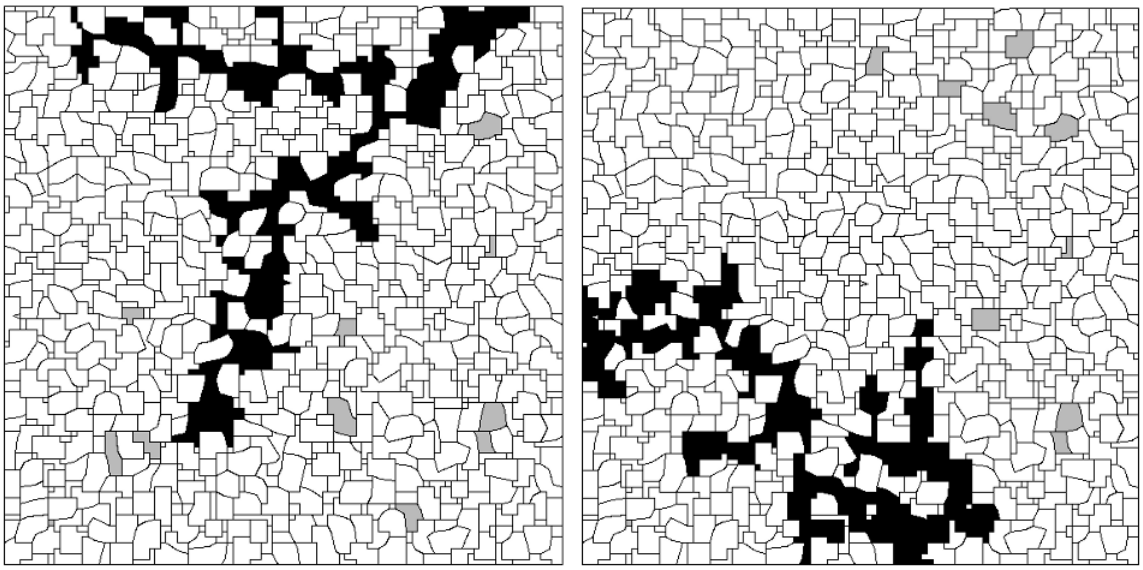


Figure 5 Two solutions for FLG9A, using a three time-period horizon and a single static patch. Stands in black are the ones selected for the old-growth patch. For simplicity, stands that are harvested in some period are shown in white and nonharvested stands in gray.

5. Final remarks

In this article, we have described a new integer programming formulation for modeling node connectivity in graphs and we have shown how it can be used to model connectivity requirements arising in forest planning models. This formulation has been tested solving a harvest scheduling model with maximum clear-cut constraints and old-growth connectivity requirements. The tests consider four variants of connectivity (combinations of rooted/unrooted and static/dynamic) on five real forest instances of medium size (352 - 1363 stands). Our computations show that (a) the linear programming relaxations of these models are both quick to solve and provide tight bounds, and (b) that in four hours of computation, we can obtain very high quality feasible solutions. Moreover, they show that rooted and static models are easier to solve when compared with unrooted and dynamic models, respectively.

The evidence that we obtain very tight linear programming relaxations suggests that specialized rounding heuristics could be an effective way of obtaining near-optimal solutions in practice. Moreover, it suggests that the use of this formulation could be practical for evaluating the performance of different heuristics by testing them on medium-sized instances.

Visual inspection of the solutions points out the necessity of including in the models presented features to promote a large interior area or lower perimeter-area ratios.

Acknowledgments

This work was partially funded by FONDECYT grants 11075028, 1085188, 1114153, and 1110674, CONICYT ANILLO grant ACT-88, Basal project CMM (Universidad de Chile), the Millenium Institute on Complex Systems (Universidad de Chile) and the Fundação para a Ciência e a Tecnologia, under the project PEst-OE/MAT/UI0152.

References

2006. Forest Stewardship Council standard. `http://www.fsc.org/`.

2008. FMOS - forest management optimization site. `http://ifmlab.for.unb.ca/fmos/`.

2010. Sustainable Forestry Initiative standard. `http://www.sfiprogram.org`.

Aldrich, M., A. Belokurov, J. Bowling, N. Dudley, C. Elliott, L. Higgins-Zogib, J. Hurd, L. Lacerda, S. Mansourian, T. McShane, D. Pollard, J. Sayer, K. Schuyt. 2004. Integrating forest protection, management and restoration at a landscape scale. Tech. rep., World Wide Fund for Nature.

Barrett, T., J. Gilless, L. Davis. 1998. Economic and fragmentation effects of clearcut restrictions. *Forest Sci.* **44**(4) 569–577.

Caro, F., M. Constantino, I. Martins, A. Weintraub. 2003. A 2-opt tabu search procedure for the multiperiod forest harvesting problem with adjacency, greenup, old growth, and even flow constraints. *Forest Sci.* **49**(5) 738–751.

Cerdeira, J., K. Gaston, L. Pinto. 2005. Connectivity in priority area selection for conservation. *Environmental Modeling and Assessment* **10**(3) 183–192.

Constantino, M., I. Martins, J. Borges. 2008. A new mixed integer programming model for harvest scheduling subject to maximum area restrictions. *Oper. Res.* **56**(3) 542–551.

Cordone, R., M. Trubian. 2006. An exact algorithm for the node weighted Steiner tree problem. *4OR* **4**(2) 124–144.

Cormen, T., C. Leiserson, R. Rivest, C. Stein. 2001. *Introduction to Algorithms*. 2nd ed. The MIT Press.

CPLEX, IBM ILOG. 2007. `http://www.ibm.com/software/integration/optimization/cplex/`.

da Cunha, A., A. Lucena, N. Maculan, M. Resende. 2009. A relax-and-cut algorithm for the prize-collecting Steiner problem in graphs. *Discrete Appl. Math.* **157**(6) 1198–1217.

Dilkina, B., C. Gomes. 2010. Solving connected subgraph problems in wildlife conservation. A. Lodi, M. Milano, P. Toth, eds., *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, *Lecture Notes in Computer Science*, vol. 6140. Springer Berlin / Heidelberg, 102–116.

Earn, D., S. Levin, P. Rohani. 2000. Coherence and conservation. *Science* **290**(5495) 1360–1364.

Espinoza, D., M. Goycoolea. 2003. EGlib, efficient general library. `http://www.dii.uchile.cl/~daespino/EGlib_doc/main.html`.

Etienne, R., J. Heesterbeek. 2000. On optimal size and number of reserves for metapopulation persistence. *J. Theor. Biol.* **203**(1) 33–50.

Fügenschuh, A., M. Fügenschuh. 2008. Integer linear programming models for topology optimization in sheet metal design. *Mathematical Methods of Operations Research* **68**(2) 313–331.

Garfinkel, R., G. Nemhauser. 1970. Optimal political districting by implicit enumeration techniques. *Management Science* **16** 495–508.

Goycoolea, M., A. Murray, F. Barahona, R. Epstein, A. Weintraub. 2005. Harvest scheduling subject to maximum area restrictions: Exploring exact approaches. *Oper. Res.* **53**(3) 490–500.

Goycoolea, M., A. Murray, J. Vielma, A. Weintraub. 2009. Evaluating approaches for solving the area restriction model in harvest scheduling. *Forest Sci.* **55**(2) 149–165.

Gustafson, E., T. Crow. 1998. Simulating spatial and temporal context of forest management using hypothetical landscapes. *Environmental Management* **22**(5) 777–787.

Keith, H., B. Mackey, D. Lindenmayer. 2009. Re-evaluation of forest biomass carbon stocks and lessons from the world's most carbon-dense forests. *Proc. Natl. Acad. Sci.* **106**(28) 11635–11640.

Kirpatric, J. 1983. An iterative method for establishing priorities for the selection of nature reserves: An example from Tasmania. *Biological Conservation* **25**(2) 127–134.

Ljubic, I., R. Weiskircher, U. Pferschy, G. Klau, P. Mutzel, M. Fischetti. 2006. An algorithmic framework for the exact solution of the prize-collecting Steiner tree problem. *Math. Programming* **105**(2-3) 427–449.

Lucena, A., M. Resende. 2004. Strong lower bounds for the prize collecting Steiner problem in graphs. *Discrete Appl. Math.* **141**(1-3) 277–294.

Luyssaert, S., E. Schulze, A. Borne, A. Knohl, D. Hessenmoller, B. Law, P. Ciais, J. Grace. 2008. Old-growth forests as global carbon sinks. *Nature* **455** 213–215.

Maculan, N., G. Plateau, A. Lisser. 2003. Integer linear models with a polynomial number of variables and constraints for some classical combinatorial optimization problems. *Pesquisa Operacional* **23**(1) 161–168.

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

Martin, R. Kipp. 1991. Using separation algorithms to generate mixed integer model reformulations. *Oper. Res. Lett.* **10**(3) 119–128.

Martins, I., J. Borges, M. Constantino. 1999. Forest management models with spatial structure constraints. Working paper Centro IO, University of Lisbon.

Martins, I., M. Constantino, J. Borges. 2005. A column generation approach for solving a non-temporal forest harvest model with spatial structure constraints. *Eur. J. Oper. Res.* **161**(2) 478–498.

McDill, M., S. Rebain, J. Braze. 2002. Harvest scheduling with area-based adjacency constraints. *Forest Sci.* **48**(4) 631–642.

Murray, A. 1999. Spatial restrictions in harvest scheduling. *Forest Sci.* **45**(1) 45–52.

Nemhauser, G., L. Wolsey. 1999. *Integer and Combinatorial Optimization*. Discrete Mathematics and Optimization, Wiley-Interscience.

Öhman, K. 2000. Creating continuous areas of old forest in long-term forest planning. *Canadian J. Forest Res.* **30**(11) 1817–1823.

Öhman, K., T. Lämas. 2005. Reducing forest fragmentation in long-term forest planning by using the shape index. *Forest Ecology and Management* **212**(1-3) 346–357.

Öhman, K., P. Wikström. 2008. Incorporating aspects of habitat fragmentation into long-term forest planning using mixed integer programming. *Forest Ecology and Management* **255**(3-4) 440–446.

Önal, H., R. Briers. 2002. Incorporating spatial criteria in optimum reserve network selection. *Proc. R. Soc. Lond. B* **269**(1508) 2437–2441.

Önal, H., R. Briers. 2003. Selection of a minimum-boundary reserve network using integer programming. *Proc. R. Soc. Lond. B* **270**(1523) 1487–1491.

Önal, H., R. Briers. 2006. Optimal selection of a connected reserve network. *Oper. Res.* **54**(2) 379–388.

Rebain, S., M. McDill. 2003a. Can mature patch constraints mitigate the fragmenting effects of harvest opening size restrictions? *Internat. Trans. Oper. Res.* **10**(5) 499–513.

Rebain, S., M. McDill. 2003b. A mixed-integer formulation of the minimum patch size problem. *Forest Sci.* **49**(4) 608–618.

ReVelle, C., R. Swain. 1970. Central facilities location. *Geographical Analysis* **2**(1) 30–42.

Segev, A. 1987. The node-weighted Steiner tree problems. *Networks* **17**(1) 1–17.

Sessions, J. 1992. Solving for habitat connections as a Steiner network problem. *Forest Sci.* **38**(1) 203–207.

Shirabe, T. 2005. A model of contiguity for spatial unit allocation. *Geographical Analysis* **37**(1) 2–16.

Thompson, E., B. Halterman, T. Lyon, R. Miller. 1973. Integrating timber and wildlife management planning. *Forestry Chronicle* **47**(49) 247–250.

Tóth, S. 2005. Modeling timber and non-timber tradeoffs in spatially-explicit forest planning. Ph.D. thesis, Forest Resources and Operations Research. Penn State University.

Tóth, S., M. McDill. 2008. Promoting large, compact mature forest patches in harvest scheduling models. *Environ. Model. and Assess.* **13**(1) 1–15.

Tóth, S., M. McDill, S. Rebain. 2006. Finding the efficient frontier of a bi-criteria, spatially explicit, harvest scheduling problem. *Forest Sci.* **52**(1) 93–107.

Vielma, J., A. Murray, D. Ryan, A. Weintraub. 2007. Improving computational capabilities for addressing volume constraints in forest harvest scheduling problems. *Eur. J. Oper. Res.* **176**(2) 1246–1264.

Williams, J. 1998. Delineating protected wildlife corridors with multi-objective programming. *Environ. Model. Assess.* **3**(1-2) 77–86.

Williams, J. 2002. A zero-one programming model for contiguous land acquisition. *Geographical Analysis* **34**(4) 330–349.

Williams, J., C. ReVelle. 1998. Reserve assemblage of critical areas: A zero-one programming approach. *Eur. J. Oper. Res.* **104**(3) 497–509.

Williams, J., C. ReVelle, S. Levin. 2005. Spatial attributes and reserve design models: A review. *Environ. Model. Assess.* **10**(3) 163–181.