The Pennsylvania State University
The Graduate School
College of Agricultural Sciences

MODELING TIMBER AND NON-TIMBER TRADE-OFFS
IN SPATIALLY-EXPLICIT FOREST PLANNING

A Thesis in
Forest Resources and Operations Research
by
Sándor F. Tóth

© 2005 Sándor F. Tóth

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2005
The thesis of Sándor F. Tóth was received and approved* by the following:

Marc E. McDill  
Associate Professor of Forest Management  
Thesis Adviser  
Chair of Committee

Wayne L. Myers  
Professor of Forest Biometrics

Terry P. Harrison  
Professor of Supply Chain and Information Systems

“Ravi” A. Ravindran  
Professor of Industrial Engineering

José A. Ventura  
Professor of Industrial Engineering

Charles H. Strauss  
Professor of Forest Economics  
Director School of Forest Resources

*Signatures are on file in the Graduate School.
ABSTRACT

Forest management is inherently a multiple-objective decision making process. Timber production, recreation, wildlife habitat and watershed protection are some of the many uses of forests. Providing the public with an optimal bundle of timber and non-timber benefits is challenging because many of these benefits conflict. Modeling the conflicting objectives, quantifying the trade-offs between them, and identifying efficient management alternatives can facilitate consensus between the stakeholders that represent the various forest uses. Spatially-explicit harvest scheduling provides an excellent modeling environment for such analyses.

This work evaluates several existing and one proposed multi-criteria mathematical programming techniques as applied to a two-, then to a three-objective spatially-explicit forest planning problem. The objectives of these models were (1) to maximize the net revenues of the forest, (2) to maximize the minimum amount of mature forest habitat in large patches that evolve across the landscape and over time given various harvest schedules, and (3), in the three-objective model, to minimize the perimeter of these patches. The comparison criteria were the number of efficient alternatives found, the time to find them, the ability of the user to filter the alternatives, and the potential of the techniques to handle n-objective problems. In both the two- and the three-objective case, the proposed Alpha-Delta and the traditional Weighted Method showed promising computational performance. It is recommended that these two methods should be employed in concert to make full use of their respective advantages.
Formulating the perimeter-minimizing criterion itself was the second objective of this study. Two 0-1 programming formulations are introduced that allow the forest planner to increase the amount of interior habitat relative to edge habitat by minimizing the boundary of the mature forest patches. Both formulations improved the shape of the patches and resulted in fewer and larger patches with more temporal overlap between them.

Finally, a procedure is introduced that strengthens the area-based adjacency constraints that restrict the size of harvest openings in spatial forest planning models. The results from test runs suggest, however, that the proposed, theoretically “better” mathematical programming formulation does not necessarily lead to shorter solution times.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>vii</td>
</tr>
<tr>
<td>List of Figures</td>
<td>viii</td>
</tr>
<tr>
<td>Acknowledgements</td>
<td>ix</td>
</tr>
<tr>
<td><strong>Chapter 1. Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>Goals</td>
<td>8</td>
</tr>
<tr>
<td>Objectives</td>
<td>9</td>
</tr>
<tr>
<td>Overview</td>
<td>11</td>
</tr>
<tr>
<td><strong>Chapter 2. Strengthening Cover Inequalities for Area-Based Adjacency</strong></td>
<td>15</td>
</tr>
<tr>
<td>Formulations of Harvest Scheduling Models</td>
<td>16</td>
</tr>
<tr>
<td>Introduction</td>
<td>28</td>
</tr>
<tr>
<td>Model formulation</td>
<td>32</td>
</tr>
<tr>
<td>Strengthening the path/cover inequalities</td>
<td>41</td>
</tr>
<tr>
<td>The strength of the extended cover inequalities</td>
<td>42</td>
</tr>
<tr>
<td>The Strengthening Algorithm</td>
<td>48</td>
</tr>
<tr>
<td>Case Studies</td>
<td>50</td>
</tr>
<tr>
<td>Results and Discussion</td>
<td>54</td>
</tr>
<tr>
<td>Conclusions</td>
<td>55</td>
</tr>
<tr>
<td>Appendix</td>
<td>55</td>
</tr>
<tr>
<td><strong>Chapter 3. Finding the Efficient Frontier of a Bi-Criteria, Spatially-explicit, Harvest Scheduling Problem</strong></td>
<td>62</td>
</tr>
<tr>
<td>Introduction</td>
<td>63</td>
</tr>
<tr>
<td>The Bi-Criteria Formulation</td>
<td>70</td>
</tr>
<tr>
<td>Methods for Identifying The Efficient Frontier of the Bi-Criteria Model</td>
<td>79</td>
</tr>
<tr>
<td>(1) The Weighted Objective Function Method (P_L)</td>
<td>80</td>
</tr>
<tr>
<td>(2) The ε-Constraining Method</td>
<td>83</td>
</tr>
<tr>
<td>(3) The Decomposition Method based on the Tchebycheff-Metric</td>
<td>86</td>
</tr>
<tr>
<td>(4) The Compromise Programming Method</td>
<td>88</td>
</tr>
<tr>
<td>(5) Hybrid Methods</td>
<td>89</td>
</tr>
<tr>
<td>(6) The Alpha-Delta Method</td>
<td>90</td>
</tr>
</tbody>
</table>
Chapter 4. Promoting large, compact mature forest patches in harvest scheduling models

Abstract: .......................................................................................................................... 105
Introduction .................................................................................................................. 106
Model Formulations ................................................................................................... 115
  The MINIMAX Model: .............................................................................................. 116
  The TOTALMIN Model: ........................................................................................... 127
Methods ....................................................................................................................... 128
Results and discussion ............................................................................................... 137
Improving the integer programming formulations ......................................................... 151
Conclusions .................................................................................................................. 153

Chapter 5. Finding the Efficient Frontier of a Tri-Criteria, Spatially-explicit, Harvest Scheduling Problem .................................................................................................................. 156
Introduction ................................................................................................................ 157
The Model Formulation............................................................................................... 165
Methods ....................................................................................................................... 177
  The Alpha-Delta Method........................................................................................... 180
  The e-Constraining Method..................................................................................... 187
  The Weighted Objective Functions and the Tchebycheff-Metric based Methods ..... 191
A case study................................................................................................................ 198
Results and Discussion ............................................................................................. 201
Conclusions .................................................................................................................. 217

Chapter 6. Conclusions ............................................................................................... 219

References.................................................................................................................... 229
List of Tables

Table 2-1. Initial age-class distribution ................................................................. 48
Table 2-2. Solution times for the 30 test problems .................................................. 51
Table 3-1. The set of efficient solutions ................................................................. 96
Table 3-2. Differences in solution optimality between the various methods .............. 97
Table 4-1. Comparison of the MINIMAX and TOTALMIN models ....................... 138
Table 4-2. The cost of perimeter minimization with the TOTALMIN model ............ 149
Table 5-1. Sub-optimal / dominated solutions found by the Tchebycheff method .... 207
List of Figures

Figure 2-1. Strengthening 2-, 3-, and 4-Way Covers .......................................................... 35
Figure 2-2. The Strengthening Algorithm ......................................................................... 43
Figure 3-1. Multi-criteria optimization terminology ......................................................... 67
Figure 3-2. The ε-Constraining Algorithm ....................................................................... 85
Figure 3-3. The Alpha-Delta Method ................................................................................ 91
Figure 3-4. The Triangles Method .................................................................................... 93
Figure 3-5. The efficient frontier of a bi-criteria harvest scheduling problem ............... 95
Figure 3-6. Cumulative solution times for each methods ................................................. 98
Figure 4-1. Mature forest habitat patches generated by the MINIMAX vs. CONTROL
          methods .................................................................................................................. 140
Figure 4-2. Mature forest patches generated by the TOTALMIN models .................... 141
Figure 4-3. Trade-offs between the NPV and the total perimeter of mature forest patches
          (TOTALMIN), and between the NPV and the maximum patch perimeter over the
          planning periods (MINIMAX) ................................................................................. 147
Figure 5-1. The first two iterations of the Alpha-Delta Algorithm with three objectives. 181
Figure 5-2. The Alpha-Delta Algorithm for $n = |P|$ objectives ..................................... 186
Figure 5-3. The ε-Constraining Algorithm for $P$ objectives ............................................ 189
Figure 5-4. The first two iterations of the ε -Constraining Algorithm with 3 objectives .. 190
Figure 5-5. The Weight Triangle ..................................................................................... 193
Figure 5-6. The Triangles Algorithm for 3 objectives ..................................................... 197
Figure 5-7. Efficient alternatives found by the ε -Constraining- and the Alpha-Delta
          methods .................................................................................................................. 202
Figure 5-8. Efficient alternatives found by the Weighted- and Tchebycheff methods .... 203
Figure 5-9. Sub-optimality and multiple-dominance ....................................................... 214
Acknowledgements

First of all and foremost, I am greatly indebted to my wife, Amy, my daughter, Éva, and my mother for their sacrifices and inspiration throughout this project. I am thankful to my academic advisor and mentor, Marc E. McDill, for inviting me to Penn State and providing me with a constant supply of academic challenges. His clear and rigorous logic was a great inspiration. The input and constructive criticism of the members of my doctoral committee, Wayne L. Myers, Terry P. Harrison, “Ravi” A. Ravindran and José A. Ventura were invaluable in the completion of this work. To these people, thank you.

Finally, I thank Marc E. McDill and the Pennsylvania Bureau of Forestry for financially supporting this research and my life at Penn State.
Chapter 1. Introduction

The forest ecosystems of the United States are one of the most valuable natural resources in the world – economically, ecologically, and spiritually. They provide economic benefits such as wood products, carbon sequestration, clean water, and employment opportunities for local communities. As the most diverse terrestrial ecosystems of the country, North American forests provide habitat for a wide range of living organisms, including humans. In addition, they provide settings for various recreational activities, including hiking, camping, fishing, hunting, canoeing, and horseback riding and they possess spiritual and aesthetic values as well.

As one would expect, these values often conflict with one another. Clearly, the more timber is harvested from a forested area, the less mature wildlife habitat will remain, the less clean water will be produced, and the less attractive the woods will be for various recreational activities. Recreation, too, can put an enormous stress on the environment. On the other hand, reducing timber production from public lands probably only shifts the impact of harvesting elsewhere, as the demand for wood products is on the rise in industrialized countries (Thomas 2000).
Conflicting relationships of this type abound in forest resource management. The multiple-objective nature of forestry, however, has not always been so prominent. For centuries, the North American woods were treated as sources of timber. The resources were vast, seemingly unlimited. Slowly but gradually, however, human encroachment and consumption have made these resources scarce and fragmented. In the meantime, the public perception of forests has changed as well. Timber production is no longer the sole, or even primary, function of forests in the eye of the public. People have become increasingly aware of the non-timber benefits that these resources have to offer. Although the shift in the public’s desire for non-timber values had been apparent for at least half a century, it was not until the 1960s that multiple-use considerations finally made their way into the legal framework of national forest management. The Multiple Use and Sustained Yield Act (1960) requires the national forests of the United States to be managed for the multiple uses of water, timber, wildlife, fish, recreation, and range (Fedkiw 1997). The National Forest Management Act (1976) further expands upon this concept and mandates forest managers to consider the many benefits of the forest explicitly in their management plans (Harrison 1983).
The legislation mentioned above does not explicitly specify how to balance the various uses of forests between the conflicting needs. This job is left to foresters. Should forest resource professionals administer surveys to chart the desires of the public, and based on these surveys, define the goals of management? Or, should they use consumption rates as measures of public desires and as targets for forest management? Public values are not only hard to measure, but they also are inconsistent and subject to constant change. What people say they need or want is not always in line with their consumer behavior. We use more timber every day, yet we also want more wilderness. Can the public expect forest resource professionals to design management regimes that reflect their diverse wants when these wants are contradictory and vaguely defined? Can the United States Forest Service and other public land management agencies have a clearly defined mission when the public’s wants are conflicting and poorly understood?

Forest planners can help the public achieve some level of consensus about forest management objectives by demonstrating the range of production possibilities to the public stakeholders (or decision makers – DM) to help them understand the pros and cons of various management alternatives. This information shows the amount of one forest benefit that would have to be forgone in order to attain a certain amount of another benefit. For
example, managers can show how much forest would have to be cut, how much wildlife
habitat would have to be lost, or what the impact would be on viewsheds in order to achieve
a given level of commodity production. Forest planners can identify and demonstrate the
trade-offs between the competing management objectives formulated by the public. With a
better understanding of these trade-offs, public stakeholder groups can reformulate or
redefine their expectations to be more in line with the production possibilities of the
resource.

Contemporary integrated forest resource management requires an understanding of
the trade-offs between competing forest uses as better management decisions can be made
based on this understanding. There has to be an ongoing dialog between forest resource
professionals and public and private stakeholders in order to make sound decisions.
Quantitative approaches that provide tangible information about the trade-offs between
various management actions can make this dialog more fruitful. The approach presented in
this dissertation to quantify the trade-offs between the competing goals involves generating
the set (or a subset) of management alternatives that are efficient (or Pareto-optimal) in the
sense that none of the objectives could be improved without compromising any of the other
objectives.
In the extreme case where there are only two management objectives and these objectives are perfectly incompatible, such as timber production versus preserving the habitat of an endangered species on a tract of forestland that can either be cut or left alone, there are only two efficient alternatives. The tract of forestland will either be cut or not cut. There is no in-between solution. If the forestland is only partially cut, less timber is produced and the habitat of the species might still be lost. In this situation, if either one of the two alternative actions is chosen, one of the objectives will be perfectly satisfied while the other one will be perfectly unsatisfied.

This situation is rare, however. There usually are many compromise alternatives in between the extreme solutions, with each one of these alternatives being optimal in the sense that none of the management objectives could be improved without taking away from at least one of the other objectives. For example, in a forest that consists of management units (polygons) many different combinations of these units could be cut in a given interval of time. If the only management objective is to maximize the profitability of timber production, then all of the units should be cut as soon as they reach their financially optimal rotation age. If, on the other hand, the only objective is to maximize the amount of mature forest patches on the landscape to provide habitat for songbirds, then none of the units
should be cut. If both of these competing objectives are present, then compromises have to be made and an ideal compromise solution should reflect the preferences of the public stakeholders or the private landowner. If these preferences are not well understood, or if consensus among the stakeholders is unlikely, then identifying all the possible efficient solutions and presenting them to the decision makers might help in finding the best compromise alternative.

The approaches proposed in this work eliminate those alternatives that are not efficient, or in other words, are dominated by other alternatives. For instance, a solution to the above bi-objective problem is dominated if at least one other solution exists that would lead to either a higher timber output without reducing the amount of forest habitat for songbirds or a greater amount of forest habitat without decreasing the timber output. Dominated solutions are clearly of no interest to the decision makers. The remaining, non-dominated or efficient solutions, however, carry very important information about the planning problem. They map out the trade-offs between the competing objectives.

The primary value of quantifying and visualizing the trade-offs is that the decision makers will have a better, more holistic understanding of the problem. Being aware of the possibilities and limits of simultaneously achieving several conflicting objectives can
facilitate consensus building. The decision makers can select the best compromise solution after having seen all or most of the alternatives and the implied trade-offs among competing objectives. Furthermore, generating an exhaustive set of efficient management alternatives, and presenting these alternatives to the decision makers might have the further advantage of protecting the forest management agency from appeals and litigation. Once the best compromise solution is agreed upon, no stakeholder can argue that other alternatives exist that had not been considered.

The proposed decision support system relies on the framework of spatially-explicit harvest scheduling. This quantitative modeling environment is an excellent tool to model both timber and non-timber objectives (e.g., Mealey and Horn 1981, Cox and Sullivan 1995, Bettinger et al. 1997, and Rebain and McDill 2003a, 2003b). As the name implies, these models can explicitly account for spatial relationships in the landscape. Since a great part of ecosystem structure and function is fundamentally spatial, this capability is essential to landscape level forest management modeling.

In summary, the multiple-objective decision system proposed in this dissertation is not intended to replace, but rather is intended to facilitate, the process of consensus building in determining desirable forest resource management strategies. Its function is to provide
quantitative information about the trade-offs between competing forest management objectives. This information can help resource professionals and public or private stakeholders better communicate during the planning process. A decision support tool like the one proposed here is particularly useful when it is not feasible to set up targets or constraints for certain forest management objectives due to the lack of understanding of resource limitations or lack of consensus.

**Goals**

The primary goal of this work is to build and evaluate a decision support tool that can provide tangible information about the trade-offs between conflicting forest management objectives. This tool is designed to work within the framework of spatially-explicit harvest scheduling – a modeling environment that is increasingly being used by forest planners. By generating and visualizing efficient (Pareto-optimal) forest management alternatives, the proposed approach will help the decision makers (1) acquire a good understanding of the structure and trade-offs of the problem, and (2), based on this knowledge, select the best compromise management scheme. Efficient approaches to generate, filter, and visualize these management alternatives will be identified.
As a secondary goal, a spatially-explicit harvest scheduling model is designed that will allow the forest planner to minimize the perimeter of mature forest patches that evolve across the landscape and over time as a result of various harvest schedules. Perimeter minimization will be used as one of the competing management objectives in evaluating the multiple-objective decision support tools proposed in this dissertation.

**Objectives**

The following research questions are addressed:

(1) The Multiple-Objective Decision Support Tools:

- What methods are available to generate a complete set of efficient solutions to a bi-objective, spatially-explicit harvest scheduling model?

- Is it possible to improve the computational performance of these methods and their ability to generate as many efficient solutions as possible?

- Is it possible to extend these algorithms to handle forest planning problems with three or more conflicting objectives?

- What is the best way to present (i.e., visualize) the set of efficient solutions to the decision maker?
• Is it possible to improve the mathematical programming formulation of area-based adjacency constraints so that the computational efficiency of the proposed multiple-objective generating techniques can be increased?

(2) The Minimum Boundary Formulation:

• Is it possible to formulate a spatially-explicit harvest scheduling model that allows the forest planner to control the shape of mature forest patches that evolve across the landscape and over time as a result of various harvest schedules?

• What effects, if any, do the minimum boundary formulations have on other spatial attributes such as the shape, temporal overlap and the number of evolving patches?

• What are the trade-offs between minimizing the perimeter of the mature forest patches and maximizing the net present value of the forest (i.e., what is the cost, in terms of net present value, of minimizing the perimeter of the mature patches?)
Overview

The dissertation consists of four chapters. Although the chapters are closely related and build on each other, each one of them can stand alone and be understood without reading the other chapters. The format and style follow the editorial guidelines of Forest Science – a leading forestry journal.

The dissertation begins with a chapter on area-based adjacency constraints (ARM). Although this work is the most abstract of the four papers and is a sidetrack in the sense that it does not address multi-objective issues, it provides the reader with an introduction to the structure and combinatorial complexity of spatially-explicit harvest scheduling problems. A procedure is proposed that attempts to improve the mathematical formulation of ARM constraints that are generated by McDill et al.’s (2002) Path Algorithm. These constraints, which restrict the size of harvest openings, have played a critical role in the development of spatially-explicit harvest scheduling models, as they require a spatially-explicit forest planning model and models with these constraints are usually very time-consuming to solve. Developing computationally efficient formulations of these models is particularly important in multiple-objective forest planning, the focus area of this dissertation, where solving a large number of combinatorial optimization problems is often necessary to generate valuable trade-off information for decision makers. A better
formulation of ARM constraints may allow analysts to identify a greater number of efficient management alternatives, and thus better describe the underlying trade-offs in a given multiple-objective forest planning problem.

Multiple-objective forest planning is the primary subject of the three remaining chapters. Various mathematical programming techniques are compared with a proposed approach based on their capability to quickly and accurately identify efficient management alternatives, first in a two-objective spatially-explicit harvest scheduling problem (Chapter 3) and then in a three-objective problem (Chapters 4 and 5). Knowing what these alternatives are can be very valuable for decision makers, as they define the trade-off between the competing objectives. However, finding a complete set of efficient alternatives is not trivial for several reasons.

First, spatially-explicit harvest scheduling models are discrete, combinatorial optimization problems where the set of efficient solutions (i.e., the efficient management alternatives) is neither convex nor continuous. This mathematical property precludes the use of many conventional, straightforward solution approaches and demands specialized algorithms. The most commonly used Weighted Objective Function Method, for instance, can only find a small, very specific subset of solutions at best. Furthermore, as different
weights on the objectives may lead to identical solutions in discrete problems, having to solve a potentially large number of integer programs only to obtain the same efficient solution can clearly result in poor computational performance. Also due to discreteness, it is not known at the outset how closely the Pareto-optimal alternatives are located in the space of achievable objective function values. This makes controlling the search space algorithmically complicated.

Another reason for the difficulty in finding efficient solutions is that spatially-explicit harvest scheduling problems can rarely be solved to full optimality due to their combinatorial complexity. Using an “optimality tolerance gap” introduces a number of numerical issues that can impact the implementation of the proposed techniques. These issues must be addressed and treated appropriately.

The harvest scheduling problem analyzed in Chapter 3 has two conflicting objectives. One is to maximize the net revenues from the forest and the other is to maximize the minimum amount over the planning horizon of mature forest habitat in large patches that evolve as a result of various harvest schedules. Chapter 4 introduces a third objective, namely to minimize the length of edge (or, perimeter) that surrounds these mature forest patches. The goal of this formulation is to allow the forest planner to control
the amount of interior mature forest habitat relative to edge habitat by increasing or
decreasing the perimeter of the patches. In addition, using the bi-objective tools discussed
in Chapter 3, the forest planner can provide the stakeholders with tangible information on
the trade-offs between any two of the three management objectives.

Chapter 5 extends the bi-objective methods to handle three or more objectives
simultaneously. The three-objective problem introduced in Chapter 4 is then analyzed
directly using the extended algorithms. Critical technical issues such as filtering and
visualizing the solutions and the underlying trade-offs in three or more objective problems
are also addressed, as well as some numerical issues related to the computational
limitations of discrete, multiple-objective optimization methods.
Chapter 2 . Strengthening Cover Inequalities for Area-Based Adjacency Formulations of Harvest Scheduling Models

By Sándor F. Tóth, Marc E. McDill and Sonney George

Abstract: A procedure is proposed to strengthen the area-restriction adjacency constraints (ARM) generated by the McDill et al.’s (2002) Path/Cover Algorithm for spatially-explicit harvest scheduling problems. The approach is similar to extending and lifting procedures for cover inequalities in 0-1 knapsack problems (Wolsey 1998). An algorithm is described that generates as many different strengthened constraints as possible from a given set of ARM constraints. The strength of the resulting constraints is discussed and the computational performance of the strengthened formulation is compared with the original Path/Cover formulation and with Goycoolea et al.’s (2005) Maximal Clique Generalized Management Unit (GMU) formulation using 300- and 500-stand hypothetical forest planning problems with three 20-year planning periods. The strengthened formulation did not outperform the original Path/Cover formulation, demonstrating that a theoretically tighter formulation does not necessarily lead to shorter solution times. However, both the
original and the strengthened Path formulations performed significantly better than the
Maximal Clique GMU approach, and, surprisingly, the 500-stand problems solved faster
than 300-stand problems.

**Additional keywords:** Integer programming, maximal cliques, extending, lifting

**Introduction**

Spatially-explicit harvest scheduling models optimize the spatial and temporal
layout of forest management actions in order to best meet management objectives such as
profit maximization, even flow of products, or wildlife habitat preservation while satisfying
a variety of constraints. They assign various silvicultural prescriptions, such as clearcut,
thinning or shelterwood harvesting treatments, to forest management units (represented as
polygons on a map, e.g., Figure 2-1) within a predetermined land-base. In addition,
spatially-explicit decisions such as road-building and reserve locations may also be
modeled. Spatially-explicit management decisions, such as whether to treat a harvest unit
or to build a road link in a given planning period, are typically modeled using 0-1 variables.
Spatially-explicit harvest scheduling models such as these are thus 0-1 programs. A variety
of restrictions, some spatially-explicit and some not, may also be modeled, including timber-flow smoothing constraints (e.g., Thompson et al. 1994), target ending age or inventory constraints (e.g., McDill and Braze 2000), and maximum harvest opening size constraints (e.g., Meneghin et al. 1988).

The need for the spatial specificity in these models, and the use of discrete optimization, has emerged primarily as a result of adjacency restrictions. Adjacency, or “green-up,” constraints limit the maximum size of contiguous harvest openings and thus are more accurately termed maximum harvest opening size constraints. These restrictions, which are often required by law or policy in North America (e.g., Boston and Bettinger 2002, American Forest & Paper Association 2000, Barrett et al. 1998), have been promoted as a tool to mitigate the negative impacts of harvesting forested ecosystems (e.g., Thompson et al. 1973, Jones et al. 1991, Murray and Church 1996a, 1996b, Snyder and ReVelle 1996a, 1996b, 1997a, 1997b, Carter et al. 1997, Murray 1999). Although maximum harvest opening size constraints do indeed disperse harvesting activities across the landscape, and thus reduce the concentration of this type of human disturbance, they have also been shown to fragment and disperse mature forest habitats (Harris 1984, Franklin and Forman 1987, Borges and Hoganson 2000, Barrett et al. 1998). To mitigate
this negative consequence of these constraints, *Rebain and McDill (2003a, 2003b)* proposed a 0-1 programming formulation that allows the forest planner to promote or to require the preservation, maintenance or creation of a certain amount of mature forest habitat in large patches over time in models with maximum harvest opening size constraints. A drawback of combining both harvest opening size constraints and mature patch habitat constraints is that the resulting models are large, complex, and hard to solve. Improving either the structure of the harvest opening size constraints or the mature patch habitat constraints can potentially make these models easier to solve. This study focuses on improving the structure of the harvest opening size constraints.

The simplest type of maximum harvest opening size constraints prevent adjacent management units from being harvested within the same time period (*McDill and Braze 2000*). This case, referred to as the Unit Restriction Model (URM, *Murray 1999*), assumes that the combined area of any two units in the forest would exceed this maximum area. The Area Restriction Model (ARM, *Murray 1999*) is more general, allowing groups of contiguous management units to be harvested concurrently as long as their combined area is less than the maximum opening size. Depending on the average area of management units, the maximum harvest opening size, and the age-class distribution of the forest, the
ARM formulation might allow for a significantly higher net present value (NPV) of the forest. Unfortunately, formulating and solving forest planning problems with ARM constraints is generally considerably more difficult than formulating and solving such problems with URM constraints.

URM constraints can be written in a number of different ways. McDill and Braze (2000) identify 16 different ways URM constraints have been formulated in the literature. They compared the computational efficiency of three of the most promising methods – Pairwise (Thompson et al. 1973), Type I-ND (Meneghin et al. 1988, Murray 1996b), and New Ordinary Adjacency Matrix (Murray and Church 1995) formulations – and found that the Type I-ND formulation was most efficient on average, although each approach performed better on a subset of the problems. The URM problem, which can be stated as selecting a subset of management units from a forest (such as Figure 2-1) for logging in such a way that no two adjacent units are cut and that the net revenues are maximized, is equivalent to the well-researched maximum weight stable set problem (SSP). The equivalence of URM and SSP is evident if one thinks of the URM as a graph with nodes representing the management units and arcs representing the adjacency relationships among these units. If the weight assigned to a node represents the net revenues that are earned if
the corresponding unit is cut, then the one-period URM problem is to identify a subset of unconnected nodes with maximum total weight. This is the maximum weight stable set problem. It is easy to generalize this equivalence to the n-period URM problem.

There are two important implications of the equivalence of URM and SSP with respect to spatially-explicit harvest scheduling models. One is that harvest scheduling models, both URM and ARM, are \( \mathcal{NP} \)-Hard, which essentially means that solution times can potentially increase with problem size faster than any polynomial function of problem size. Both the URM and the ARM are \( \mathcal{NP} \)-Hard, because the ARM is a generalization of the URM, and the URM is equivalent to the SSP, which is known to be \( \mathcal{NP} \)-Hard (Nemhauser and Trotter 1974). The other implication is related to the following result (see the proof in the Appendix), which allows the construction of a set of very strong adjacency constraints for URM problems. These constraints are called maximal clique inequalities.

**Claim 1.** Given a graph \( G = (V, E) \) with \( n = |V| \), and a set of incidence vectors of the stable sets \( X = \{x \in B^n : x_i + x_j \leq 1 \text{ for } e = (i, j) \in E\} \), the clique inequality \( \sum_{j \in C} x_j \leq 1 \) is (1) valid\(^1\) and (2) defines a facet of \( \text{conv}(X) \), where a clique \( C \subseteq V \) is a maximal set of nodes with the property that for all pairs \( i, j \in C \), the edge \( e = (i, j) \) is in \( E \) (Wolsey 1998).

---

\(^1\) An inequality \( \pi x \leq \pi_0 \) is valid for \( X \in B^n \) if \( \pi x \leq \pi_0 \) holds for all \( x \in X \) (based on Wolsey 1998).
 Claim 1 applies to URM problems regardless of how the adjacency of the management units is defined. If two units are defined as adjacent based on point adjacency or proximity, then $|C|$ can be any positive integer. If, on the other hand, the adjacency of two units is based on a shared common boundary, then $|C| \leq 4$ (Haken and Appel 1976).

The facet-defining property of maximal clique inequalities in URM problems has been mentioned in Murray and Church (1996a) and was later utilized by Goycoolea et al. (2005) and Murray et al. (2004) in solving ARM problems. It is important to point out, however, that the maximal clique constraints are facet-defining only in the feasible region of the adjacency constraints. In the more general harvest scheduling problem, where many other spatial and non-spatial restrictions are modeled, maximal cliques might not be facet-defining at all.

In contrast to the URM, ARM problems were initially deemed impossible to formulate in a linear model (Murray 1999) and only heuristics were employed to solve them (e.g., Lockwood and Moore 1993, Caro et al. 2003, or Richards and Gunn 2003).

However, McDill et al. (2002) identified two exact, linear, 0-1 programming formulations of the ARM. The first formulation uses constraints that are designed to allow groups of

---

2 The facet-defining property refers to the strength of valid inequalities. See the Appendix for a precise definition.
contiguous management units to be harvested as long as their combined area does not exceed the maximum harvest opening limit. McDill et al. (2002) present an algorithm, which they call the Path Algorithm, that recursively enumerates all sets of contiguous management units whose combined areas just exceed the maximum allowable harvest level. The constraints created this way are similar to cover inequalities in 0-1 knapsack problems. However, the overall structure of the ARM is much more complex: it is a combination of several, overlapping knapsacks where adjacency restrictions are imposed upon the items to be placed in the knapsacks. Furthermore, while in the knapsack case the cover inequalities can be derived directly from one ‘parent’ constraint of form \( \sum_{j=1}^{n} a_j x_j \leq b \ (x \in B^n) \), in forest planning problems, the adjacency constraints have to be enumerated from a graph, or adjacency table. Due to the similarities between McDill et al.’s (2002) path constraints and the knapsack cover inequalities, we refer to the path constraints as cover constraints or cover inequalities in this paper. The advantage of the path/cover formulation over the one discussed next is that it does not require the introduction of additional 0-1 decision variables.

McDill et al.’s (2002) second formulation uses separate variables for each possible combination of contiguous management units within the forest whose total area does not
McDill et al. (2002) refer to these combinations as Generalized Management Units (GMU). With this formulation, the same types of adjacency constraints as those used in URM models (e.g., pairwise or clique) can be written on the set of GMUs. McDill et al. (2002) used pairwise constraints in their initial experiment, but they observe that Type 1 constraints (Meneghin et al. 1988), which are similar to the clique constraints in the stable set problem, might also be used with the expectation that this would likely result in a tighter formulation and reduced solution times. This was expected because in the URM case Type 1 formulations had already been found to perform generally better than the most promising alternatives (McDill and Braze 2000).

Gooycoolea et al. (2005) applied maximal clique constraints to GMUs to formulate ARM problems. Their study found that the Maximal Clique GMU approach performed better than McDill et al.’s (2002) Path/Cover formulation in terms of computational efficiency.

Crowe et al. (2003) appended what they call “ARM clique constraints” to McDill et al.’s (2002) cover inequalities, arguing that the “clique” concept can be applied to ARM models if the total area of a mutually adjacent set of management units exceeds the maximum opening size. These constraints are of form \[ \sum_{j \in C} a_j x_j \leq A_{\text{max}} \], where \( C \) is a maximal clique of management units for which \( \sum_{j \in C} a_j > A_{\text{max}} \) holds (\( x_j \) is a binary decision variable).

23
variable which takes a value of 1 if unit $j$ is cut and 0 otherwise, $a_j$ is the area of unit $j$, and $A_{\text{max}}$ is the maximum harvest opening size). Crowe et al.'s (2003) “clique constraints” are, in fact, knapsack constraints, and it would be more appropriate to call them ARM knapsack constraints, rather than clique constraints. Clique constraints normally involve coefficients that are ones. In any case, Crowe et al. (2003) found that the appended formulation did not outperform McDill et al.'s (2002) cover/path approach. This is not surprising since the proposed ARM cliques are, in fact, generally weaker than the path/cover formulation.

Consider, for example, an ARM clique constraint: $X = \{15x_1 + 10x_2 + 25x_3 \leq 40\}$. This constraint is redundant since $\text{conv}(X) = \{0 \leq x_i \leq 1, \ x_1 + x_2 + x_3 \leq 2, \ x_i \in R \}$ where $i = 1, 2, 3$. In other words, the non-negativity constraints, the upper bounds and the cover constraint, $x_1 + x_2 + x_3 \leq 2$, dominate the ARM clique. Redundant constraints such as $15x_1 + 10x_2 + 25x_3 \leq 40$ can hinder the solution process because they destroy the 0-1 coefficient structure of the cover constraint matrix if they do not get eliminated during pre-processing.

However, there are cases when Crowe et al.'s (2003) ARM cliques can help better approximate the ARM’s integral convex hull. Using their example, the ARM clique constraint $X = \{16x_1 + 14x_3 + 17x_4 + 17x_6 \leq 40\}$ cuts off fractional solutions such as
\( x_1 = x_3 = x_4 = x_6 = 2/3 \) from the feasible set defined by McDill et al.’s (2002) cover constraints: \( \{x_1 + x_3 + x_4 \leq 2, x_1 + x_3 + x_6 \leq 2, x_1 + x_4 + x_6 \leq 2, x_3 + x_4 + x_6 \leq 2 \} \). The ARM clique constraint, however, does not dominate the cover constraints either, as it allows fractional solutions such as \( x_1 = x_3 = 1, x_4 = 10/17, x_6 = 0 \), which the cover constraints do not. Thus, both the covers and the ARM clique are needed to better approximate \( \text{conv}(X) \) in this case. It should be noted, however, that one can construct an even stronger valid inequality, namely, \( x_1 + x_3 + x_4 + x_6 \leq 2 \) that dominates both the covers and the ARM clique constraints (inequality \( x_1 + x_3 + x_4 + x_6 \leq 3 \), incorrectly labeled by Crowe et al. (2003) as invalid, is also dominated). This paper discusses how such constraints might be developed from the set of covers.

Gunn and Richards’ (2005) so-called “Stand-Centered” constraints can also be an alternative to or complement McDill et al.’s (2002) cover inequalities, because they can cut off fractional solutions from the feasible set defined by the covers. To illustrate this, consider Gunn and Richards’ (2005) example: a forest that consists of five stands with acreages of \( a_1 = 10\text{ha}, a_2 = 20\text{ha}, a_3 = 25\text{ha}, a_4 = 15\text{ha}, \) and \( a_k = 15\text{ha} \), where stands 1 and 2, 1 and 4, and 3 and 4 are adjacent and stands 1 through 4 are all adjacent to stand \( k \),
and the maximum harvest area is 40 ha. The following cover constraints can be written using the Path Algorithm: \( C = \{x_{1t} + x_{2t} + x_{4t} \leq 2, \ x_{1t} + x_{3t} + x_{4t} \leq 2, \ x_{1t} + x_{kt} + x_{3t} \leq 2, \ x_{2t} + x_{kt} + x_{3t} \leq 2, \ x_{2t} + x_{4t} + x_{kt} \leq 2, \ x_{3t} + x_{4t} + x_{kt} \leq 2\} \), where \( x_{it} \in \{0,1\} \) is equal to 1 if stand \( i \) is to be cut in period \( t \), and 0 otherwise for \( i = 1, 2, 3, 4, k \). For the same set of management units, the strengthened stand-centered constraints are

\[
SC = \{15x_{4t} + 10x_{kt} + 10x_{3t} \leq 20, \ 10x_{1t} + 20x_{2t} + 25x_{3t} + 15x_{4t} + 30x_{kt} \leq 60, \\
10x_{1t} + 10x_{kt} + 25x_{3t} + 10x_{4t} \leq 35, \ 5x_{1t} + 20x_{2t} + 10x_{kt} + 15x_{4t} \leq 35\}.
\]

Since constraint

\[
10x_{1t} + 20x_{2t} + 25x_{3t} + 15x_{4t} + 30x_{kt} \leq 60
\]

cuts off fractional solutions such as

\[
x_{1t} = x_{2t} = x_{3t} = x_{4t} = x_{kt} = 2/3 \quad \text{from} \ C, \quad \text{and cover} \ x_{2t} + x_{4t} + x_{kt} \leq 2, \quad \text{in turn, cuts off}
\]

\[
x_{2t} = x_{kt} = 1, \ x_{1t} = x_{2t} = 0, \ x_{4t} = 1/3 \quad \text{from} \ SC, \quad \text{both the covers and the stand-centered constraints are needed to better approximate the convex hull of the ARM.}
\]

*Gunn and Richards' (2005)* formulation is attractive, because (1) the number of stand-centered constraints needed is equal to the number of units in a forest, which is much less than the number of covers that might be needed, and (2) unlike finding the covers, generating stand-centered constraints does not require a potentially very time-consuming recursive enumeration.
One way to utilize Crowe et al.’s (2003) ARM cliques and Gunn and Richards’ (2005) stand-centered constraints would be to append them, on the fly, and on an individual basis, to McDill et al.’s (2002) cover constraints if and only if they cut off the optimal fractional solution to an LP sub-problem during the Branch and Bound Algorithm.

This paper presents a procedure that strengthens the cover inequalities introduced as path constraints by McDill et al. (2002). This strengthening procedure generates very strong inequalities for ARM problems. Although we postulate that these inequalities define some of the facets of the integral convex hull of the ARM adjacency constraints, we only prove this property for one specific case.

The rest of this paper is organized as follows. The next section introduces the integer programming formulation of a simple harvest scheduling problem with McDill et al.’s (2002) path/cover constraints. The structure of these constraints, as generated by the Path Algorithm, will be discussed next, followed by a description of the procedure that strengthens them. The computational efficiency of the strengthened constraints was assessed by formulating 30 simple harvest scheduling problems in 3 ways: (1) with the basic cover inequalities of McDill et al. (2002), (2) with Goycoolea et al.’s (2005) Maximal Clique GMUs, and (3) with the strengthened cover inequalities proposed in this paper. The
discussion section assesses the potential of applying the new procedure to forest planning problems with various spatial structures. The paper concludes with a discussion of the pros and cons of the cover approach versus the Maximal Clique GMU procedure.

**Model formulation**

The general structure of the spatially explicit ARM model, where the adjacency constraints are generated by the Path Algorithm, is as follows:

\[
\text{Max} Z = \sum_{m=1}^{M} \sum_{t=1}^{T} c_{mt} \cdot A_{mt} \cdot x_{mt} 
\]

Subject to:

\[
\sum_{t=0}^{T} x_{mt} \leq 1 \quad \text{for } m = 1, 2, \ldots, M 
\]

\[
\sum_{m=1}^{M} v_{mt} \cdot A_{mt} \cdot x_{mt} - H_{t} = 0 \quad \text{for } t = 0, 1, 2, \ldots T 
\]

\[
b_{t+1}H_{t} - H_{t+1} \leq 0 \quad \text{for } t = 0, 1, 2, \ldots T-1 
\]

\[
b_{t+1}H_{t} - H_{t+1} \leq 0 \quad \text{for } t = 0, 1, 2, \ldots T-1 
\]

\[
\sum_{j \in C_{t}} x_{jt} \leq |C_{t}| - 1 \quad \forall C_{t} \text{ and for } t = 0, 1, 2, \ldots T 
\]

\[
\sum_{m=1}^{M} \sum_{t=0}^{T} \left( (Age_{mt} - \bar{Age}) \cdot A_{mt} \cdot x_{mt} \right) \geq 0
\]

\[
x_{mt} \in \{0, 1\} \quad \text{for } m = 1, 2, \ldots, M \text{ and } t = 0, 1, 2, \ldots T 
\]
where \( x_{mt} \) = a binary variable whose value is 1 if management unit \( m \) is to be harvested in period \( t \) for \( t = 1, 2, \ldots, T \); when \( t = 0 \), the value of the binary variable is 1 if management unit \( m \) is not harvested at all during the planning horizon (i.e., \( X_{m0} \) represents the “do-nothing” alternative for management unit \( m \)),

\[ M = \text{the number of management units in the forest}, \]

\[ T = \text{the number of periods in the planning horizon}, \]

\[ c_{mt} = \text{the net discounted revenue per hectare if management unit } m \text{ is harvested in period } t, \]

\[ A_m = \text{the area of management unit } m \text{ in hectares}, \]

\[ v_{mt} = \text{the volume of sawtimber in } \text{m}^3/\text{ha} \text{ harvested from management unit } m \text{ if it is harvested in period } t, \]

\[ H_t = \text{the total volume of sawtimber in } \text{m}^3 \text{ harvested in period } t, \]

\[ b_{l,t} = \text{a lower bound on decreases in the harvest level between periods } t \text{ and } t+1 \]

(where, for example, \( b_{l,t} = 1 \) would require non-declining harvests and \( b_{l,t} = 0.9 \) would allow a decrease of up to 10%).
\( b_{h, t} = \) an upper bound on increases in the harvest level between periods \( t \) and \( t+1 \)

(where \( b_{h, t} = 1 \) would allow no increase in the harvest level and \( b_{h, t} = 1.1 \) would allow an increase of up to 10%),

\( C_i = \) the set of indexes corresponding to the management units in cover \( i \),

\( Age^T_{mt} = \) the age of management unit \( m \) at the end of the planning horizon if it is harvested in period \( t \); and

\( Age^T = \) the target average age of the forest at the end of the planning horizon.

Equation (1) specifies the objective function of the problem, namely to maximize the discounted net revenue from the forest during the planning horizon plus the forest value of each stand at the end of the planning horizon. The first set of constraints (2) are logical constraints. They require a management unit to be assigned to at most one prescription, including a do-nothing prescription. Harvest variables \( X_m \) are only created for periods where the stand is old enough to be harvested (i.e., it is older in that period than the predefined minimum rotation age). The second set of constraints (3) are harvest accounting constraints. They sum the harvest volume for each period and assign the resulting value to the harvest (continuous) variables \( H_t \). Constraint sets (4) and (5) are flow constraints. They limit the rate at which the harvest volume can increase or decrease from one period to
the next. Constraint set (6) represents the maximum harvest opening constraints as minimal
covers generated by the Path Algorithm. These constraints assume that the exclusion
period equals one planning period, i.e., that once a management unit, or group of
contiguous units, has been harvested, no adjacent management units can be harvested until
at least one period has passed. The structure of these constraints is easy to generalize to
alternative exclusion periods which are integer multiples of a planning period (see, for
example, Snyder and ReVelle 1997b). Constraint (7) is an ending age constraint. It
requires the average age of the forest at the end of the planning horizon to be at least
\( \bar{Age} \) years, preventing the model from over-harvesting the forest during the planning
horizon. Constraint (8) identifies the management unit treatment alternative variables as
binary.

Constraint sets (2) and (6) must be replaced in order to formulate the ARM using
Goycoolea et al.’s (2005) Maximal Clique GMU approach. Constraint set (2) is replaced
by the following:

\[
\sum_{\omega \in G_u, \ t=0} \sum_{\tau} \chi_{ut} \leq 1 \quad \text{for } m = 1, 2, ..., M_0
\]
where \( M_O \) = the set of original management units (note that in this case \( M \) represents the complete set of management units, including GMUs), and

\[ G_m = \text{the set of management units in } M \text{ that include original management unit } m. \]

Constraint set (6) is replaced with the following:

\[
\sum_{m \in K_j} x_{mt} \leq 1 \quad \text{for all } K_j \text{ and } t = 1, 2, ..., T \tag{10}
\]

where \( K_j \) = the set of indexes corresponding to the set of GMUs that contain at least one stand in maximal clique \( j \) of the original management units. A maximal clique of the original management units is a set of those units that are mutually adjacent; it is maximal if and only if there is no other management unit that is adjacent to all of the units in the clique.

**Strengthening the path/cover inequalities**

As discussed above, harvest scheduling models with ARM constraints are \( \mathcal{NP} \)-Hard even without complicating constraints such as ones that ensure a smooth flow of timber production (constraint sets (3), (4), and (5)) or a desirable ending forest age or inventory (constraint (7)). Realistic, on-the-ground applications of the forest planning problem are
large scale (e.g., 500-2,500 management units in many cases). As a result, obtaining optimal solutions through exact optimization within a reasonable time frame is currently beyond our reach – at least for problems of the above scale. Still, this study focuses on how an exact integer programming formulation of the ARM may be improved with the hope that the combination of improvements in formulations, exact solution algorithms, and computing power will someday allow us to obtain exact solutions to such problems in a reasonable time. In spite of the current computational limitations, research in exact spatial optimization for forest planning problems is justified for several reasons: (1) there is a rapid, ongoing improvement in both the hardware and software computational technologies that allow us to solve larger and larger problems every day, (2) cheap and highly sophisticated off-the-shelf solver packages can conveniently be used to solve forest planning problems in standard IP format, and (3) the quality (optimality) of the solutions can be controlled. Furthermore, the performance of heuristic approaches in finding good feasible solutions cannot be evaluated without exact solutions (Murray et al. 2004).

In order to strengthen McDill et al.’s (2002) path formulation, the structure of the path/cover constraints must be understood first. For the sake of simple notation, we assume that there is only one planning period. The binary decision variable $x_j$ takes the value 1 if
management unit $j$ is cut, and it takes 0 otherwise. The minimal path/cover inequalities generated by McDill et al.’s (2002) Path Algorithm are of form $\sum_{j \in C_i} x_j \leq |C_i| - 1$, for any $i$, where $C_i$ is a set of contiguous management units for which $\sum_{j \in C_i} a_j > A_{\text{max}}$ holds ($a_j =$ area of unit $j$, and $A_{\text{max}} =$ maximum harvest limit), but $\sum_{j \in C_i \setminus \{l\}} a_j \leq A_{\text{max}}$ for any $l \in C_i$.

Set $C_i$ can be called a “path,” as in McDill et al. (2002), or, using the analogy with the cover inequalities that arise in 0-1 Knapsack problems, it can be called a “cover” (Wolsey 1998). The term “cover” stems from the fact that the combined area of the management units that appear in a constraint exceeds, or equivalently, covers the allowable contiguous harvest limit. They are minimal because if any one unit is excluded from $C_i$, the total area of the remaining management units will be less than the harvest limit. For example in Figure 2-1, the set of management units $\{13, 14, 43, 50\}$ is a minimal cover, because the combined area of these four units exceeds the maximum opening size (48 ha), but if any one of these units is removed from the cover, the remaining units do not form a cover. (In the figure, the first number in each polygon is the management unit ID number and the second number is the initial age-class of the management unit. Areas are also identified for some units. Twenty-year age-classes are used in this example, so age-class number 1 refers to the 0-20 year age-class, 2 refers to the 21-40 year age-class, and so on.)
Minimal covers can be classified based on the number of decision variables (management units) they comprise. For example, in Figure 2-1, \{24, 38\}, \{18, 31, 40\}, and \{13, 14, 43, 50\} are 2-, 3-, and 4-way covers, corresponding to inequalities \(x_{24} + x_{38} \leq 1\), \(x_{18} + x_{31} + x_{40} \leq 2\), and \(x_{13} + x_{14} + x_{43} + x_{50} \leq 3\), respectively. Throughout the rest of this
paper, it is assumed that each management unit in the forest has an area less than or equal to the allowable contiguous cut limit. Thus, \(|C_i| \geq 2\), for any \(i\).

Let \(C\) denote the set of all possible minimal covers that arise from a certain forest planning problem. Further, let \(s\) denote the index of a management unit that is adjacent to \(2 \leq k \leq |C_i|\) management units in cover \(C_i\). This management unit is a candidate for being included in cover \(C_i\), forming an extended cover \(E_s(C_i)\). \(S_{C_i}\) denotes the set of all management units that are candidates for being included in cover \(C_i\), \(\^C_i^k\) is the set of all possible \(l\)-combinations of the \(k\) units in \(C_i\) that are adjacent to candidate unit \(s\), and, finally, let \(\^C_i^k\) denote one such combination \((1 \leq l \leq k)\). In other words, \(\^C_i^k\) is a set of \(l\) management units that are all members of \(C_i\) and all adjacent to candidate unit \(s\) (\(\^C_i^k\) is the set of all possible sets of \(\^C_i^k\)).

**Proposition 1** (Extension): Cover inequality \(\sum_{j \in C_i} x_j \leq |C_i| - 1\) can be extended with unit \(s\) to \(\sum_{j \in E_s(C_i)} x_j \leq |C_i| - 1\), where \(E_s(C_i) = C_i \cup \{s : x_s + \sum_{j \in \^C_i^k} x_j \leq k - 1\}\) for all \(\^C_i^k \in \^C_i^k\). Extended cover inequality \(\sum_{j \in E_s(C_i)} x_j \leq |C_i| - 1\) is valid (see proof in Appendix).

To illustrate the use of **Proposition 1**, consider the 4-way cover, \(\{13, 14, 43, 50\}\), in the 50-stand hypothetical forest planning problem shown in Figure 2-1. The maximum allowable contiguous harvest area is 48 ha, so management unit 3 can be included in cover
{13, 14, 43, 50}, because \( x_{50} + x_{14} + x_3 \leq 2 \), \( x_{50} + x_{43} + x_3 \leq 2 \), and \( x_{43} + x_{14} + x_3 \leq 2 \) all hold, since \{3, 14\}, \{3, 43\}, \{3, 50\} are all covers. The resulting inequality

\[ x_{13} + x_{43} + x_{50} + x_{14} + x_3 \leq 3 \]

is valid, because if there was an integer solution for which

\[ x_{13} + x_{43} + x_{50} + x_{14} + x_3 > 3 \]

held, then either the original 4-way cover

\[( x_{13} + x_{43} + x_{50} + x_{14} \leq 3 )\], or some of the 2-way covers, \(( x_3 + x_{14} \leq 1, x_3 + x_{43} \leq 1, \text{ or} x_3 + x_{50} \leq 1)\), or both, would be violated as well. In other words,

\[ x_{13} + x_{43} + x_{50} + x_{14} + x_3 \leq 3 \] does not cut off any integer solutions from the feasible region defined by the original covers. We call the set of management units represented by the decision variables on the left hand side of this inequality, \{13, 43, 50, 14, 3\} an “extended cover” of cover \{13, 14, 43, 50\}.

To illustrate cases when Proposition 1 does not hold, consider units 17 or 29 in Figure 2-1 that are both elements of set \( S_{C_i} \), as both are adjacent to two members of the original cover \{13, 14, 43, 50\}. These units cannot be included in the cover, because

neither \( x_{13} + x_{17} \leq 1 \) and \( x_{50} + x_{17} \leq 1 \) nor \( x_{14} + x_{29} \leq 1 \) and \( x_{43} + x_{29} \leq 1 \) hold, as cutting \{13, 17\}, \{50, 17\}, \{14, 29\} or \{43, 29\} do not violate the maximum harvest opening restriction.
Now, notice that if certain conditions hold, the extended cover constraint
\[ \sum_{j \in E_s(C_i)} x_j \leq |C_i| - 1 \]
may be further strengthened by either lifting the coefficient of \( x_s \) from 1 to \( \alpha_s \leq k - 1 \), or by adding another management unit from set \( S_{C_i} \) to the cover constraint.

**Proposition 2 (Lifting):** Lifting the coefficient of \( x_s \) to
\[ \alpha_s = k - \min \left\{ l : x_s + \sum_{j \in c_i} x_j \leq l \text{ for all } c_i \in C_i \right\} \]
leads to valid inequality
\[ \sum_{j \in C_i} x_j + \alpha_s x_s \leq |C_i| - 1 \] (see proof in Appendix).

To illustrate **Proposition 2**, consider again the 4-way cover, \{13, 14, 43, 50\}, in Figure 2-1. After including management unit 3 in the cover (using **Proposition 1**), the coefficient of \( x_3 \) can be lifted from 1 to \( \alpha_3 = 3 - 1 = 2 \), as each element (1-combination) of set \{14, 43, 50\} forms a 2-way minimal cover with management unit \( s \). In other words, unit \( s \) cannot be harvested together with any of the adjacent units in the cover without violating the maximum harvest opening restriction. This yields inequality
\[ x_{13} + x_{43} + x_{50} + x_{14} + 2x_3 \leq 3 \] (Note: if management unit 3 was small enough to give rise to a 3-way cover, such as \{3, 14, 28\}, the coefficient of \( x_3 \) could not be lifted). The lifted inequality is valid, because for any integer solution for which \( x_{13} + x_{43} + x_{50} + x_{14} + 2x_3 > 3 \) either \( x_{13} + x_{43} + x_{50} + x_{14} > 3 \), or at least one of the following constraints must hold:
In either case, at least one of the original cover constraints is violated.

Finally, consider a situation where, for a given cover $C_i$, there exists two or more management units for which the condition in Proposition 1 holds. Let $Q_i \subseteq S_i$ denote the unique set of management units for cover $C_i$ such that each unit in $Q_i$ passes Proposition 1. We call $Q_i$ the “cover extension set” for $C_i$. The question is which subset or subsets of $Q_i$ can be included together in a single extended cover constraint, lifted or not, without violating the original constraint. To illustrate this situation, consider the 3-way cover constraint, $x_{18} + x_{31} + x_{40} \leq 2$, in Figure 2-1. Three management units, $\{6, 8, 15\}$ satisfy the condition in Proposition 1. Therefore, the cover extension set, $Q_i$, for minimal cover $\{18, 31, 40\}$ is $\{6, 8, 15\}$. Clearly, including all three units simultaneously in a single extended cover constraint, i.e., $x_{18} + x_{31} + x_{40} + x_6 + x_{15} + x_8 \leq 2$, is not valid because including all three candidates in the cover would not permit cutting units 6, 8, and 18 or units 15, 8, and 40 simultaneously, even though these harvest schedules would not violate the maximum harvest opening restriction. Looking at subsets of 2, neither 6 and 8 nor 8 and 18 can be included simultaneously in a single extended cover constraint for the same reasons listed above. However, management units 6 and 18 can be combined in a single
extended cover constraint, as this would strengthen the constraint without preventing any legitimate harvest combinations. Thus, two separate extended cover constraints need to be constructed: \( x_{18} + x_{31} + x_{40} + x_0 + x_{15} \leq 2 \), and \( x_{18} + x_{31} + x_{40} + x_0 \leq 2 \). We use the term “compatible” to describe subsets of \( Q_{c_i} \) that can be included together in a single extended cover constraint. The issue of compatibility is complicated by the fact that it may be possible to lift the coefficients of some of the units in the cover extension set when they are included singly in an extended cover constraint, but these units may only be compatible with other units in the cover extension set if their coefficient is not lifted.

Although it is possible to construct rules for various spatial configurations, on a case by case basis, a parsimonious set of rules that determines which subsets of the cover extension set, lifted or not, can be included in a single extended cover constraint without making it invalid remained elusive within this study. It is suspected that obtaining cuts that are as strong as possible requires the maximization of the sum of coefficients corresponding to the decision variables that are to be included in the cover while maintaining the validity of the constraint. A separate extended cover constraint should be written for each of the compatible subsets that is not dominated by another.
The strength of the extended cover inequalities

Integer programming formulations that provide tight initial dual bounds on the optimal integer feasible solution are often easier to solve than those with less tight bounds. Using constraints that better approximate the integral convex hull of the problem can increase the chances of obtaining tighter dual bounds and, thus, can lead to shorter solution times. Ideally, for the ARM-based spatially-explicit harvest scheduling problem, each extended cover constraint would define a facet of the convex hull, and as many of them would be found as possible.

The question naturally arises: how strong are the extended and lifted cover constraints in the ARM? Are they facet-defining?

**Proposition 3** (The Strength of the Proposed Inequalities): The extending and lifting procedure defined by Proposition 1 and 2 yields facet-defining inequalities for $\text{conv}(X_C)$ of form $\sum_{j \in C_i} x_j + \alpha s, x_s \leq |C_i| - 1$, where $x_j, x_s \in B^n$ for $\forall j$, if $|Q_C| = 1$ (see proof in Appendix).

Again, although the extended and lifted inequalities define some of the facets of $X_C$, they do not necessarily define the facets of the convex hull of the general harvest scheduling problem that has more constraints than solely adjacency restrictions.
The Strengthening Algorithm

The Strengthening Algorithm attempts to generate as many strong inequalities from the initial set of minimal covers as possible. The goal is to tightly approximate $\text{conv}(X)$. The flowchart in Figure 2-2 illustrates the key steps of the algorithm.

The Strengthening Algorithm starts with selecting a cover $C_i$ from the complete set of minimal covers $C$, generated by the Path Algorithm (McDill et al. 2002). The set of management units that are adjacent to at least two units in $C_i$, denoted by $S_i$, is identified next. One management unit is selected from this set and tested for extension using Proposition 1. If the management unit fails the test, its decision variable is discarded from set $S_i$ and another one is selected. If the management unit passes the test, the Lifting Test is applied to the unit. Using Proposition 2, the maximum value of the coefficient $\alpha_s$, that can be assigned in an extended cover constraint to the decision variable ($x_s$) corresponding to the management unit $s$ is determined, and $x_s$ enters the set $E(C_i)$. Once all of the units in $S_i$ have been processed, a branching process called the Branching Subroutine is called. The Branching Subroutine determines, on a case by case basis, which combinations of management units of set $Q_{C_i}$, lifted or not, are compatible.

Step 1: First, a management unit, say unit $s$ is taken from $Q_{C_i}$.
Step 2: The Branching Subroutine creates as many nodes as the number of integer values $\alpha_s$ (the coefficient of $x_s$ in the extended cover constraint) can take. For example, if
the maximum value of $\alpha_i$ is 3, then 3 nodes are created. At Node 1, $\alpha_s$ is set to 1, at Node 2, it is set to 2, and so on. In other words, the coefficient of the decision variable that can enter the cover is lifted to all integer values it can possibly take without invalidating the cover inequality $C_i$.

**Step 3:** Starting from one of the nodes (say, Node $\Phi$), another management unit is selected from $Q_{C_i}$ and tested for compatibility with $\alpha_s x_s$ (Compatibility Test). If the new management unit is compatible with $\alpha_s x_s$, the maximum value of its coefficient is calculated, using **Proposition 2**, and the branching process is repeated from Step 2. In other words, new nodes are created that branch out from Node $\Phi$. The new nodes correspond to the integer values that the coefficient of the variable corresponding to the new management unit can take without creating an invalid extended cover constraint.

**Step 4:** The algorithm recursively iterates until no more management units are left to process from set $Q_{C_i}$, i.e., no more lifting or extension is possible starting from Node $\Phi$. The incumbent constraints are saved at the end of each branch originating from Node $\Phi$.

**Step 5:** Each remaining node corresponding to a stand in $Q_{C_i}$ and a possible lifted coefficient associated with the stand is evaluated in the same way as Node $\Phi$. The incumbent constraints are saved at the end of each final branch.
Step 6: Since it is possible to generate identical constraints by following two separate paths in the branching tree, the redundant constraints are eliminated from the final set of extended and lifted covers.

After the Branching Subroutine is complete, another minimal cover is selected from $C$ and the strengthening process starts all over again. The Strengthening Algorithm terminates when no more minimal covers remain in $C$. As a final step, the redundant constraints that are generated by processing different covers are eliminated.

The following example illustrates the Strengthening Algorithm and the Branching Subroutine. Consider the minimal cover $C_{\text{example}} = \{18, 31, 40\}$ again (Figure 2-1). After passing the Extension Test (see Proposition 1), management units 6, 8 and 15 each belong to the cover extension set, i.e., $Q_{\text{example}} = \{6, 8, 15\}$. The lifting test determines that $\alpha_6 = \alpha_8 = \alpha_{15} = 1$ (see Proposition 2). In the Branching Subroutine, three nodes are created, one for each management unit, since the only non-zero value that the coefficients on the variables representing the units in $Q_{\text{example}}$ can take in an extended cover constraint is 1.

Starting with Unit 6, set $\alpha_6 = 1$ at the first node. Unit 8 fails the compatibility test with Unit 6. They are incompatible, because if both of them were included in a single extended cover constraint, the right hand side of the resulting inequality,
\( x_6 + x_{18} + x_{31} + x_{40} \leq 2 \), would be violated at \( x_6 = x_8 = x_{18} = 1 \), which is a feasible combination of harvests. Unit 15, however, is compatible with Unit 6, since no more than two units of the set \{6, 15, 18, 31, 40\} can be cut in the same period (Figure 2-1). There are no more units left in \( Q_{\text{convex}} \), so this branch can be fathomed, yielding valid inequality

\[ x_6 + x_{15} + x_{18} + x_{31} + x_{40} \leq 2 . \]

Starting with Unit 8 (\( \alpha_8 = 1 \)), we find that neither Unit 6 nor Unit 15 is compatible with Unit 8. Thus, this branch has to be fathomed with valid inequality

\[ x_8 + x_{18} + x_{31} + x_{40} \leq 2 . \] Note that the compatibility of Unit 6 and 8 has already been tested at the first node, so it is only Unit 8 and 15 that need to be evaluated at this point.

Finally, starting with Unit 15 (\( \alpha_{15} = 1 \)), we find that only Unit 6 is compatible with Unit 15. However, extended cover inequality \( x_6 + x_{15} + x_{18} + x_{31} + x_{40} \leq 2 \) has already been generated at the first node. Therefore, this branch also has to be fathomed because of redundancy.

Both the Strengthening Algorithm and the Branching Subroutine generate a number of redundant (dominated) constraints. Building these constraints and then eliminating them increases the computational expense of the procedure. Thus, minimizing the redundancies and fine-tuning the algorithm is critical to the success of the proposed approach.
Finally, notice that if the Path Algorithm and the strengthening procedure are applied to a URM problem, the resulting extended covers would be maximal cliques (see **Claim 1**). As the combined area of any two adjacent units in the URM exceeds the harvest opening limit, the Path Algorithm would yield only pairwise constraints (i.e., 2-way minimal covers). As a result, the coefficient of a candidate unit for which the condition in **Proposition 1** holds (i.e., this unit is a member of the cover extension set, $Q_{c_i}$), cannot be lifted. In other words, the maximum value of the coefficient that corresponds to the candidate unit would be 1, because it can only be adjacent to the two units in the cover (if it was adjacent to only one of the units, it would not be a candidate). Notice also that any two or more units in $Q_{c_i}$ can only be compatible, or, equivalently, can only be included in the extended cover, if they are mutually adjacent. Thus, the extended covers, formed by compatible subsets of $Q_{c_i}$ and the original minimal cover are maximal cliques. This is an important observation for three reasons. First, it establishes a structural link between the maximal clique constraints in URM problems and the extended cover inequalities in the ARM. Second, it implies that the Strengthening Algorithm would serve as a potential tool to generate maximal clique constraints for URM problems. Third, as the maximal clique formulation has been identified as the most efficient formulation of the maximum weight
stable set problem, it is legitimately expected that the proposed procedure yields a computationally superior formulation of the ARM as well.

**Case Studies**

In order to evaluate the computational efficiency of the strengthened cover inequalities, twenty-four 300-stand and six 500-stand hypothetical forest planning problems were constructed based on the model described in the ‘Model Formulation’ section. Each problem had one forest type and one site class. The initial age-class distribution of each forest mimics a typical Pennsylvania hardwood forest (Table 2-1). As the hypothetical forests comprise different spatial configurations of management units and the acreage of the individual units is predefined, the actual percentages of the age-classes might deviate slightly from the figures in the table.

<table>
<thead>
<tr>
<th>Age-classes</th>
<th>Total Area (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0-20</td>
<td>20</td>
</tr>
<tr>
<td>2 21-40</td>
<td>3</td>
</tr>
<tr>
<td>3 41-60</td>
<td>11</td>
</tr>
<tr>
<td>4 61-80</td>
<td>30</td>
</tr>
<tr>
<td>5 81-100</td>
<td>24</td>
</tr>
<tr>
<td>6 101-120</td>
<td>12</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>
The planning horizon of the models was 60 years (three 20-year planning periods). The minimum rotation age was 60, and the optimal rotation age, based on maximizing the land expectation value (LEV), was 80 years. The four possible prescriptions were to cut the management unit in period 1, period 2, or period 3, or not at all. A maximum harvest opening size of 48 ha was imposed, and adjacent units were allowed to be harvested concurrently as long as their combined area is less than this maximum opening size. All management units must be smaller than the maximum harvest opening size. The average age of the forest at the end of the planning horizon was constrained to be at least 30 years – half of the minimum rotation age.

The 30 test problems were solved using CPLEX 9.0 (ILOG CPLEX 2003) on a dual processor Intel® XEON™ CPU 3.06 GHz computer with 3.0 GB RAM under a Windows platform (Microsoft Windows XP Professional Version 2002, Service Pack 1). The relative MIP gap tolerance parameter (optimality gap) was set to $5.0 \times 10^{-4}$ (0.05%); the MIP variable selection strategy parameter was set to ‘4’ (pseudo-reduced cost-based); the MIP parallel threads parameter was set to “2” (i.e., parallel processing); and the working memory limit was set to 448KB.
Each problem was formulated three different ways, using (1) Goycoolea et al.’s (2005) Maximal Clique GMUs, (2) McDill et al.’s (2002) path/cover constraints, and (3) the strengthened covers introduced in this paper. The sole criterion of comparison was the solution time in seconds needed to find the first integer feasible solution within the 0.05% predefined optimality gap.

Results and Discussion

Table 2-2 shows the solution times for each formulation of the 30 test problems. The Maximal Clique GMU was the fastest in solving six of the problems; the original path/cover formulation solved 13 problems in the least time; and the strengthened cover formulation resulted in the shortest solution time for 11 problems. On average, both the original and the strengthened path/cover formulations performed significantly better than the Maximal Clique GMU approach. It appears that the benefits of the strong maximal cliques are offset by the fact that the GMU formulation requires more binary decision variables and more constraints than the other two formulations. It should be noted, however, that the GMU approach can account for fixed cost savings, such as timber sale,
road building, fencing and harvesting costs, by jointly managing a cluster of management units (McDill et al. 2002). Neither of the path/cover formulations have this capability.

### Table 2-2. Solution times for the 30 test problems

<table>
<thead>
<tr>
<th>300 Stand Problems</th>
<th>Solution Time (seconds)</th>
<th>500 Stand Problems</th>
<th>Solution Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GMU</td>
<td>PATH</td>
<td>STPATH</td>
</tr>
<tr>
<td>1</td>
<td>1,212.81</td>
<td>1,713.52</td>
<td>6,534.64</td>
</tr>
<tr>
<td>2</td>
<td>6,836.09</td>
<td>320.22</td>
<td>1,762.89</td>
</tr>
<tr>
<td>3</td>
<td>21,504.89</td>
<td>6,341.00</td>
<td>4,950.45</td>
</tr>
<tr>
<td>4</td>
<td>153,032.89</td>
<td>2,492.28</td>
<td>942.61</td>
</tr>
<tr>
<td>5</td>
<td>10,690.70</td>
<td>1,146.19</td>
<td>962.97</td>
</tr>
<tr>
<td>6</td>
<td>917.02</td>
<td>1,477.83</td>
<td>1,474.39</td>
</tr>
<tr>
<td>7</td>
<td>21,504.89</td>
<td>6,341.00</td>
<td>4,950.45</td>
</tr>
<tr>
<td>8</td>
<td>12,195.59</td>
<td>1,007.78</td>
<td>754.92</td>
</tr>
<tr>
<td>9</td>
<td>21,186.91</td>
<td>810.64</td>
<td>8,005.08</td>
</tr>
<tr>
<td>10</td>
<td>2,745.80</td>
<td>344.78</td>
<td>368.03</td>
</tr>
<tr>
<td>11</td>
<td>6,479.27</td>
<td>4,619.72</td>
<td>9,357.36</td>
</tr>
<tr>
<td>12</td>
<td>617.00</td>
<td>1,082.81</td>
<td>1,543.09</td>
</tr>
<tr>
<td>13</td>
<td>2,116.41</td>
<td>642.86</td>
<td>992.63</td>
</tr>
<tr>
<td>14</td>
<td>9,310.34</td>
<td>141.03</td>
<td>905.38</td>
</tr>
<tr>
<td>15</td>
<td>2,860.92</td>
<td>4,465.53</td>
<td>2,954.33</td>
</tr>
<tr>
<td>16</td>
<td>8,876.11</td>
<td>513.84</td>
<td>2,522.42</td>
</tr>
<tr>
<td>17</td>
<td>11,067.83</td>
<td>5,214.91</td>
<td>3,351.16</td>
</tr>
<tr>
<td>18</td>
<td>28,238.77</td>
<td>2,532.66</td>
<td>2,894.55</td>
</tr>
<tr>
<td>19</td>
<td>1,680.97</td>
<td>1,923.22</td>
<td>1,391.31</td>
</tr>
<tr>
<td>20</td>
<td>7,937.50</td>
<td>367.75</td>
<td>17,154.55</td>
</tr>
<tr>
<td>21</td>
<td>1,991.55</td>
<td>6,425.05</td>
<td>17,133.69</td>
</tr>
<tr>
<td>22</td>
<td>907.19</td>
<td>245.81</td>
<td>850.89</td>
</tr>
<tr>
<td>23</td>
<td>1,964.83</td>
<td>2,362.33</td>
<td>1,547.36</td>
</tr>
<tr>
<td>24</td>
<td>754.33</td>
<td>1,053.25</td>
<td>1,920.00</td>
</tr>
<tr>
<td>Average (mins)</td>
<td>221.12</td>
<td>34.50</td>
<td>60.45</td>
</tr>
<tr>
<td>STDEV (mins)</td>
<td>510.28</td>
<td>32.32</td>
<td>76.08</td>
</tr>
</tbody>
</table>

**Abreviations:**
- ACD = Age Class Distribution
- PATH = McDill et al.'s (2002) path/cover approach
- STPATH = Strengthened path/cover constraints
- P. Periods = Planning periods
- STDEV = Standard Deviation

Although the strengthened cover constraints were shown to be facet-defining on $X_C$ if $|Q_C| = 1$, the test runs demonstrate that a tighter formulation does not necessarily lead to shorter solution times (Table 2-2). The strengthened covers did not outperform the
original covers in terms of computational efficiency. We suspect that generating the strengthened constraints and adding them to the formulation up front might make the linear programming sub-problems harder to solve. This, in turn, can lead to longer overall solution times. There are a couple of ways to overcome this problem. One is to strengthen only those cover inequalities that are binding at a given linear programming iteration. Adding “cuts” this way would lead to a cutting plane or branch-and-cut algorithm that is specific to the spatially-explicit harvest scheduling polytope. Alternatively, the set of these cuts could be generated up front, but they would only be added to the problem if they are violated by the optimal fractional solution to the linear programming sub-problem at a given iteration. Future research will tell if these alternatives work better.

A surprising finding of the test runs is that the 500-stand problems solved faster than the 300-stand problems (Table 2-2). Although the very small sample size used in this study (only six 500 stand problems were solved) does not allow for general conclusions, this result suggests that additional work is needed to examine the relationship between problem size (number of management units) and computational complexity. Such a research could test the hypothesis that when the forest contains more management units, the
corresponding integer program becomes less “lumpy” and more closely approximates a continuous optimization problem.

Finally, it is useful to consider how the spatial structure of forest planning problems can impact the number of extension and lifting opportunities in a path/cover-based formulation. While many cover constraints may be extended and lifted in some problems, none will be possible in others. The more cover constraints can be strengthened in a problem, the more likely it is to reduce solution times. The average size of the management units relative to the maximum contiguous harvest area and the frequency of adjacencies per unit are the major factors that determine the strengthening potential of the proposed procedure. The smaller the average size of the units relative to the cut limit, the less likely it is that the cover inequalities can be strengthened. This is because a spatial structure that has smaller units relative to the cut limit will give rise to covers with more members. Longer covers are more difficult to strengthen because smaller units that are adjacent to the cover are less likely to satisfy Proposition 1 and 2. Similarly, a higher frequency of adjacencies is likely to increase the number of candidate stands that are adjacent to more than one member of the cover. Thus, lifting opportunities are more likely to occur when there are more adjacencies. Clearly, a close inspection of the spatial structure of the
problem in question is needed to decide if the strengthening procedure should be employed at all.

**Conclusions**

This paper has presented a method for strengthening the path/cover constraints, generated by McDill et al.’s (2002) Path Algorithm for area-based forest planning models. The strengthened constraints were shown to be facet-defining for at least some cases. Although one would expect that the theoretically tighter formulation would lead to shorter solution times, this did not occur in the test runs. This leaves open the question of what is the “best” formulation to use for solving forest planning models with ARM constraints. Our comparison of three approaches suggests that, as long as it is not critical to account for fixed cost savings, the original path approach is the best. However, we did not compare these approaches with Crowe et al.’s (2003) ARM cliques and Gunn and Richards’ (2005) stand-centered constraints. It is possible that some combination of cover constraints, strengthened cover constraints, ARM cliques, or stand-centered constraints would provide superior computational results. Alternatively, ARM cliques, stand-centered constraints, and strengthened covers could be used in a cutting plane approach where they would be
appended to the list of original covers on an individual basis if they cut off fractional linear programming solutions at the nodes of the branch and bound algorithm. This cutting plane approach might speed up the solution process by eliminating certain fractional solutions from consideration without making the linear programming sub-problems significantly more difficult.

This study clearly demonstrated that both the original path/cover and the strengthened cover formulations outperform the Maximal Clique GMU approach in terms of computational efficiency. The latter approach, however, can identify fixed cost savings, such as timber sale and harvesting costs, by jointly managing several smaller units. The price of this benefit is a higher number of decision variables and constraints, and thus longer solution times. Among the three formulations tested here, we recommend using the Maximal Clique GMU approach if fixed costs are critical to the planning problem, and the path/cover approach otherwise.

Appendix

Definition 1 (Valid Inequalities). An inequality \( \pi x \leq \pi_0 \) is valid for \( X \in B^\pi \) if \( \pi x \leq \pi_0 \) holds for all \( x \in X \) (based on Wolsey 1998).
Definition 2 (Linear Independence). Vectors (directions) $x^2 - x^1, ..., x^k - x^1$ are linearly independent if and only if system \[
\sum_{j=1}^{k-1} \mu_j (x^{j+1} - x^1) = 0
\]
has no solution other than $\mu_j = 0$ for all $j = 1,...,k$ ($\mu_j$ are unknown scalars).

Definition 3 (Affine Independence). The points $x^1, ..., x^k \in R^n$ are affinely independent if the $k-1$ directions $x^2 - x^1, ..., x^k - x^1$ are linearly independent, or alternatively the $k$ vectors $(x^1,1),..., (x^k,1) \in R^{n+1}$ are linearly independent (Wolsey 1998).

Definition 4 (Dimension of polyhedra). The dimension of polyhedron $X$, denoted $\dim(X)$, is equal to the maximum number of affinely independent points in $X$ less one (Wolsey 1998).

Definition 5 (Full-dimensional polyhedra). $X \subseteq R^n$ is full-dimensional if and only if $\dim(X) = n$ (Wolsey 1998).

Definition 6 (Facet-defining Property of Valid Inequalities). For full-dimensional polyhedra $X \subseteq Z^n$, valid inequality $\pi \cdot x \leq \pi_0$ defines a facet of $\text{conv}(X)$ if and only if there exist $n$ affinely independent points in $X (x^1, ..., x^n \in X)$ satisfying $\pi \cdot x = \pi_0$ (Wolsey 1998).
Proof of Validity for Claim 1. See Definition 1. Let $X_p$ and $X_c$ denote the feasible regions defined by the original incidence vectors (pair-wise inequalities) and the proposed clique inequalities respectively. We show that if a point $x^T \not\in X_c$ then $x^T \not\in X_p$ either. If $x^T \not\in X_c$, then $\sum_{j \in C} x_j^T \geq 2$ must be true. Thus, there must be at least one pair of nodes $(i, j) \in C$ such that $x_i + x_j \geq 2$. And, since any edge in $C$ is also in $E$ (by problem statement), at least one of the original pair-wise inequalities is violated. ■

Proof of the facet-defining property for Claim 1. Taking $X = \{ x \in B^n : x_i + x_j \leq 1 \text{ for } e = (i, j) \in E \}$, we have that $\dim(\text{conv}(X)) = n$, because the maximum number of affinely independent points in $X$ is $n+1$. These points are $x^0 = 0$, $x^1 = e_1$, $x^j = e_j$, and $x^n = e_n$, where $e_j$ is the $j$th unit vector. Thus, $X$ is full-dimensional. Now, we consider the valid inequality $\sum_{j \in C} x_j \leq 1$, where all pairs of $i, j \in C$, the edge $e = (i, j) \in E$, and show that it is facet-defining. See definitions 2-6.

The following $n$ integral feasible points satisfy $\sum_{j \in C} x_j \leq 1$ at equality. For simplicity, the first $|C|$ coordinates of each of the following points correspond to the indices of the nodes in $C$. The first $|C|$ points are: $x^j = (e_j)$ for $j = 1, \ldots, |C|$, where $e_j$ is the $j$th unit
vector. The remaining \( n - |C| \) points are: \( x' = (e_j + e_i) \) for \( j \neq i \) and \( (i, j) \notin E \). Observe that the number of pairs \( (i, j) \) such that \( j \neq i \) and \( (i, j) \notin E \) is greater than or equal to \( n - |C| \). For if not, \( C \) would not be a maximal clique. As \( x' = (e_j + e_i) \) lies on the hyperplane \[ \sum_{k=1}^{n} \mu_k x_k = \mu_0 \] (where \( \mu_k \) and \( \mu_0 \) are unknown scalars), \( \mu_j = \mu_0 \). As \( x' = (e_j + e_i) \) lies on the hyperplane \[ \sum_{k=1}^{n} \mu_k x_k = \mu_j, \mu_j + \mu_i = \mu_j, \text{ so } \mu_i = 0. \] So the generic hyperplane is \[ \sum_{j \in C} \mu_j x_j = \mu_j \text{ with solutions } (\mu_j) = \lambda (1), \text{ where } \lambda = \mu_j \neq 0. \] Thus, \( \sum x_j \leq 1 \) is facet-defining.

**Proof of Proposition 1** (Extension). Let \( X \) denote the feasible set of integer solutions defined by \( C \) (the set of minimal covers). We show that if integer solution \( x^T \) does not satisfy \( \sum_{j \in E(C)} x_j \leq |C_j| - 1 \), then \( x^T \notin X \). If \( \sum_{j \in E(C)} x_j > |C_j| - 1 \), then either \( \sum_{j \in C_i} x_j > |C_i| - 1 \) or \( \sum_{j \in C_j} x_j > k - 1 \) for some \( z^{k^j} \in z^{k^j} \) or both, must be true. In any case, at least one cover inequality of the original set of covers is violated.

**Proof of Proposition 2** (Lifting). Following the same argument as in the proof of **Proposition 1**, we show that if an integer solution \( x^T \) violates the lifted constraint \[ \sum_{j \in C_i} x_j + \alpha_s x_s \leq |C_i| - 1, \text{ then } x^T \notin X \]. If \( \sum_{j \in C_i} x_j + \alpha_s x_s > |C_i| - 1 \), then either
\[ \sum_{j \in C_i} x_j^T > |C_i| - 1 \text{ or } x_s + \sum_{j \in C_i^k} x_j^T > l \text{ (where } l = k - \alpha_s \text{) for some } c_i^k \in C_i^k, \text{ or both, must hold. In any case, at least one cover inequality of the original set of covers is violated.} \]

**Proof of Proposition 3** (Strength of cover constraints if \(|Q_{C_i}| = 1\)). Taking

\[ X = \left\{ \sum_{j \in C_i} x_j \leq |C_i| - 1 \text{ for each } C_i \in C \right\}, \text{ we have that } \dim(\text{conv}(X)) = n, \text{ because the maximum number of affinely independent points in } X \text{ is } n+1. \text{ These points are} \]

\[ x^0 = 0, x^1 = e_1, ..., x^d = e_s, \text{ and } x^n = e_n, \text{ where } e_i \text{ is the } i^{th} \text{ unit vector. Thus, } X \text{ is full-dimensional. Now, we consider the valid inequality } \sum_{j \in C_i} x_j + \alpha_s x_s \leq |C_i| - 1, \text{ where } x_s \in Q_{C_i}, \text{ and show that it is facet-defining. See definitions 2-6.} \]

The following \( n \) integral feasible points satisfy \( \sum_{j \in C_i} x_j + \alpha_s x_s \leq |C_i| - 1 \) at equality.

For simplicity, the first \(|C_i|\) coordinates of each of the following points correspond to the indices of the management units that are in the minimal cover \( C_i \), and the \((|C_i| + 1)^{th}\) coordinate corresponds to \( x_s \). The first \(|C_i| + 1\) points are: \( x^1 = (u_{k^{th}}, 0) \), \( x^2 = (u_k, 0) \) \( \ldots \), \( x^{|C_i|} = (u_{k^{th}}, 0) \), and \( x^{|C_i|+1} = (v, e_1) \) where \( u_k \) is vector of length \(|C_i|\) with the \( k^{th}\) element being 0 and all the other elements are 1s, 0 is a zero vector of length \( n - |C_i| \) and \( e_k \) is the \( k^{th}\) unit vector (of length \( n - |C_i| \)). Vector \( v \) (of length \(|C_i|\)) is the zero vector if \( \alpha_s = |C_i| - 1, v = e_s \) if \( \alpha_s = |C_i| - 2, \) where \( k \) is the index of the management unit which unit
s is not adjacent to. If \( 1 \leq \alpha_s < |C_i| - 2 \) then \(|C_i| - 1 - \alpha_s\) number of 1s have to be assigned to a subset of the first \(|C_i|\) positions in such a way that the point remains feasible. The remaining \( n - |C| - 1 \) points are: \( x^m = (u_l, e_k) \) where \( l \in \{1, 2, ..., |C_i|\} \) and \((u_l, e_k)\) is a feasible solution. Observe that the number of feasible pairs \((u_l, e_k)\) is greater than or equal to \( n - |C| - 1 \). For if not, \( Q_i \geq 2 \) and a stronger inequality for \( C_i \) would be possible.

Now, observe that the position of the one zero element within the first \(|C_i|\) coordinates for the first \(|C_i|\) points is different for each of these points. This implies that each of the \(|C_i| - 1\) directions, pointing from \( x^1 \) to the rest of the \(|C_i|\) points will have one (1) element at a position where all the other directions have 0. This means none of these \(|C_i| - 1\) directions can be expressed as a linear combination of the other directions. The \(|C_i|^{th}\) direction, \( x^{F_i + 1} - x^1 \), has a 1 at the \((|C_i| + 1)^{th}\) position, where all of the other directions have a zero element (they were constructed this way). Thus, the \(|C_i|^{th}\) direction cannot be expressed as a linear combination of the other directions either, no matter what vector \( v \) looks like. Points \( x^{F_i + 1}, x^{F_i + 2}, \ldots, x^n \) have the same coordinates at the first \(|C_i| + 1\) positions as \( x^1 \) does (again, they were constructed this way), and they have only one (1) element, at a position that is different for each of them. Therefore, none of the
directions pointing from $x^1$ to these points can be expressed as a linear combination of the rest of the directions. Thus, the $n$ points are affinely independent.

**Illustration of the proof to Proposition 3.** Consider the extended and lifted cover inequality $x_{13} + x_{14} + x_{43} + x_{50} + 2x_3 \leq 3$ in Figure 2-1. This cover inequality was constructed from minimal cover $x_{13} + x_{14} + x_{43} + x_{50} \leq 3$ using Proposition 1 and 2. We show that $x_{13} + x_{14} + x_{43} + x_{50} + 2x_3 \leq 3$ defines a facet of $conv(X)$, where $X \in B^n$ ($n = 50$ as the forest in Figure 2-1 consists of 50 management units). The following 50 feasible points of form $x^i = (x_{13}, x_{14}, x_{43}, x_{50}, x_3, \ldots)$ satisfy $x_{13} + x_{14} + x_{43} + x_{50} + 2x_3 \leq 3$ at equality:

$$
x^1 = ((1,1,0,1,0), 0), \quad x^2 = ((1,0,1,0,0), 0), \quad x^3 = ((0,1,1,1,0), 0), \quad x^4 = ((0,1,1,1,0), e^4), \\
x^5 = ((0,0,1,1,0), 0), \quad x^6 = ((1,0,1,1,0), e^3), \ldots, \quad x^4 = ((1,0,1,1,0), e^{45}), \ldots, \\
x^{50} = ((1,0,1,1,0), e^{45}).
$$

After generating the directions from $x^1$ to the remaining 49 points, system $\sum_{j=1}^{49} \mu_j (x^{j+1} - x^1) = 0$ has no solution other than $\mu_j = 0$ for $\forall j = 1, \ldots, 50$. Thus, the 49 directions are linearly independent, which implies that the 50 points are affinely independent.
Chapter 3  Finding the Efficient Frontier of a Bi-Criteria, Spatially-explicit, Harvest Scheduling Problem

Sándor F. Tóth, Marc E. McDill, and Stephanie Rebain

Abstract: This paper evaluates the performance of five traditional methods and one new method of generating the efficient frontier for a bi-criteria, spatially-explicit harvest scheduling problem. The problem is to find all possible efficient solutions, thus defining the trade-offs between two objectives: (1) maximizing the net present value of the forest and (2) maximizing the minimum area over the planning horizon in large, mature forest patches. The methods for generating the efficient frontier were tested using a hypothetical forest consisting of 50 stands. The methods were compared based on the number of efficient solutions each method can identify and on how quickly the solutions were identified. The potential to generalize these algorithms to three- or n-criteria cases is also assessed. Three of the traditional approaches, the \( \varepsilon \)-constraining-, the Triangles method, the decomposition algorithm based on the Tchebycheff metric, and the new, proposed method are capable of generating all or most of the efficient solutions. However, the Triangles and
the new method far outperformed the other approaches in terms of solution time. The new method, called Alpha-Delta, appears to be the simplest to generalize to the tri-criteria case.

**Key words**: Multi-criteria optimization, wildlife habitat, trade-off analysis, 0-1 programming

**Introduction**

Society expects more from its forest resources than merely timber production. Increasingly, values such as wildlife habitat, recreation, water quality, aesthetics and spiritual values are also recognized. In accordance with these expectations, the Multiple-Use Sustained-Yield Act (1960) requires the national forests of the United States to be managed for the multiple uses of water, timber, wildlife, fish, recreation, and range (Fedkiw 1997). The emerging field of multiple-objective forest planning is a promising tool, given this diverse nature of forest resources management (Pukkala 2002). Sustaining large patches of mature forests (forest stands that are older than a certain age) throughout the planning horizon can contribute to fulfilling many of the multiple uses demanded by society. In addition, adjacency constraints, which limit the size of harvest openings, have been promoted as contributing to these objectives (e.g. Thompson et al. 1973, Jones et al.)
Murray and Church 1996a, 1996b, Snyder and ReVelle 1996a, 1996b, 1997a, 1997b, Carter et al. 1997, Murray 1999). However, adjacency constraints tend to work against the goal of developing and preserving large, mature patches of forest (Harris 1984, Franklin and Forman 1987, Rebain and McDill 2003a,b). As adjacency constraints are intended to prevent large clearcuts, they tend to disperse harvesting activities across the forest. Large, contiguous tracts of mature forests are not likely to be maintained this way.

One way of tackling this problem is to include constraints that require the models to maintain a minimum total area in mature patches meeting a minimum age and size requirement, while maximizing the net present value (NPV) of the forest (Rebain and McDill 2003a,b). However, it might be difficult to identify an appropriate total area of large, mature patches that is adequate to meet conservation goals but not overly restrictive. Nevertheless, single-objective models have often been applied to forest planning problems with multiple objectives where the minimum or maximum level of other outputs or values were defined by constraints (Leuschner et al. 1975, Mealey and Horn 1981, Cox and Sullivan 1995, Bettinger et al. 1997). *A priori* methods, such as goal programming (Field 1973, Kao and Brodie 1979, Field et al 1980, Arp and Lavigne 1982, Hotvedt 1983, Mendoza 1987, Rustagi and Bare 1987, or Davis and Lui 1991) also suffer from the
limitation that the decision maker (DM) is required to identify his or her preferences prior to the solution process. Expecting the DM to specify the desired level of achievement or to specify his or her preferences for the various objectives without knowing what is possible is not only unrealistic but might also lead to poor management decisions. An interactive method, where the DM helps drop certain regions of the feasible solution set by comparing and ranking a limited number of alternative solutions, is a feasible approach that might remedy this shortcoming. With an interactive approach, at each iteration the DM progressively articulates his or her preferences and the focus of the search becomes more confined. This way, the search converges toward a solution that maximizes the DM’s utility – the best compromise solution. The major drawback of the interactive approach is that it requires an active and possibly lengthy involvement of the DM. Still, in cases with three or more criteria, the interactive approach might be the only viable option, since the complete set of alternatives and the trade-offs among them are usually too difficult for the DM to visualize, let alone to analyze and rank. Shin and Ravindran (1991) and Miettinen (1999) provide comprehensive discussions of these interactive methods.

With a bi-objective model such as the one discussed in this paper the DM can be spared this potentially lengthy interaction and need not define his or her preferences until
the potential solution alternatives are identified. This approach allows the DM to explore all possible trade-offs between the two objectives – in this case, the net present value of the forest and the minimum area over time in large, mature patches. This approach provides the DM with a more holistic understanding of the trade-offs and more alternatives to choose from. This way, the DM can select his or her preferred alternative after having seen all possible solutions. This type of approach is called an ‘a posteriori’ approach in the operations research (OR) literature (Miettinen 1999).

Thus, when objectives conflict, as in the spatially-explicit harvest scheduling problem discussed in this paper, it might be useful to determine the set of Pareto Optimal, or efficient, solutions – i.e., the potential management alternatives. An efficient solution (such as Point E in Figure 3-1), as opposed to a dominated solution (such as Point C in the figure), occurs when it is not possible to increase the attainment of one objective without reducing the attainment of another. Knowing the set of efficient solutions can help the DM understand the trade-offs between the competing objectives.

In a multi-objective optimization problem, the level of achievement of each objective defines each axis of the objective space (Figure 3-1). Because the problems in this paper are mixed-integer programming (MIP) problems, the set of attainable objective
values, which can be represented in this space, is not a convex set. In fact, it is not a continuous set; it consists of a set of discrete points corresponding to the potentially large, but finite number of feasible solutions such as Points A, B, C, D, and E in Figure 3-1. The fact that this set is not convex requires us to distinguish between supported and non-supported Pareto optimal solutions. A series of weighted objective functions, where weights are assigned to each of the problem objectives and summed to obtain a single objective function value, can be used to identify the corner points of the convex hull of the efficient solution set, such as Points A and B in Figure 3-1. These points are commonly called supported strong (or strict) Pareto optima (T’kindt et al. 2002). Efficient solutions that are not on the border of the convex hull, such as Point E in Figure 3-1, are called non-
supported strict Pareto optima. Such optima will not be identified by a weighted objective function approach.

The set of strong Pareto optima, both supported and non-supported, define the outside (convex) corners of a line called the efficient frontier or trade-off curve. Points on the vertical or horizontal line segments between these corners may represent dominated solutions, such as Point D in Figure 3-1. However, there does not necessarily exist a solution at every point on these line segments due to the integer nature of the problem. Solutions on these line segments, such as the one represented at Point D, are called weak Pareto optima. The efficient frontier separates the region where additional efficient solutions are known not to exist from the region where dominated solutions may exist. Knowing the efficient frontier can be valuable to decision makers because it demonstrates the possible trade-offs between the objectives of a given problem.

When only two objectives are of interest, 2-dimensional efficient frontiers can be generated to describe the trade-offs between these objectives. These curves can help determine which forest management plans will result in the best combination of achievements with respect to each goal. Importantly, trade-off curves allow the DM to assess the amount of one goal that must be given up in order to achieve a given increase in
the amount of another goal. Trade-off curves for forest and wildlife management problems have been presented in Roise et al. (1990), Holland et al. (1994), Cox and Sullivan (1995), Arthaud and Rose (1996), Church et al. (1996) and (2000), Snyder and Revelle (1997a), Williams (1998), and Richards and Gunn (2000). Cohon et al. (1979) developed a technique for generating an approximate or exact representation of the efficient frontier for convex bi-criteria problems.

This research addresses the question of how to identify the efficient frontier as efficiently as possible for spatially-explicit harvest scheduling models where the set of solutions is not convex in the objective space. Efficiency is important because the procedure for identifying all possible efficient solutions can be computationally expensive (Miettinen 1999). This is especially true with spatially-explicit models such as the one used in this paper. These models are typically formulated as mixed-integer programming (MIP) problems, which are, in general, \(\mathcal{NP}\)-Hard. Essentially this means that solution times can potentially increase with problem size faster than any polynomial function of problem size. Wolsey (1998) provides a more precise definition of the \(\mathcal{NP}\)-Hard property. This paper tests the performance of five traditional methods and one proposed method of
generating the efficient frontier for a bi-criteria, spatially-explicit harvest scheduling problem.

The Bi-Criteria Formulation

This section describes the formulation of the example spatially-explicit harvest scheduling model. It includes harvest flow constraints, maximum harvest opening size constraints, constraints that define the minimum area of large, mature patch habitat over time, and a minimum average ending age constraint. The model formulation of the mature forest patch criterion is essentially the same as the one presented in Rebain and McDill (2003a). Formulation of the maximum harvest area constraints is a generalization of the formulation presented in McDill et al. (2003a).

\[
Max Z = \sum_{m=1}^{M} A_m [c_{m0}X_{m0} + \sum_{t=h_m}^{T} c_{mt}X_{mt}]
\]  

(1)

\[
Max \lambda
\]  

(2)

subject to:

\[
X_{m0} + \sum_{t=h_m}^{T} X_{mt} \leq 1 \quad \text{for } m = 1, 2, \ldots M
\]  

(3)
\[
\sum_{m \in M_t} v_{mt} \cdot A_m \cdot X_{mt} - H_t = 0 \quad \text{for } t = 1, 2, \ldots T \tag{4}
\]

\[
b_{lt} H_t - H_{t+1} \leq 0 \quad \text{for } t = 1, 2, \ldots T-1 \tag{5}
\]

\[
-b_{lt} H_t + H_{t+1} \leq 0 \quad \text{for } t = 1, 2, \ldots T-1 \tag{6}
\]

\[
\sum_{m \in P_t} X_{mt} \leq n_{P_t} - 1 \quad \text{for all } P_t \in P \text{ and } t = h_t, \ldots, T \tag{7}
\]

\[
\sum_{m \in P_t} X_{mj} - O_{mt} \geq 0 \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \tag{8}
\]

\[
\sum_{m \in S_t} O_{mt} - n_c B_{ct} \geq 0 \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots, T \tag{9}
\]

\[
\sum_{m \in C_t} B_{ct} - BO_{mt} \geq 0 \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \tag{10}
\]

\[
\sum_{m=1}^{M} A_m BO_{mt} \geq \lambda \quad \text{for } t = 1, 2, \ldots, T \tag{11}
\]

\[
\sum_{m=1}^{M} A_m [(Age_{0t}^T - \bar{Age}^T_t)X_{0t} + \sum_{t=h_m}^{T} (Age_{mt}^T - \bar{Age}^T_t)X_{mt}] \geq 0 \tag{12}
\]

\[
X_{mt}, O_{mt}, BO_{mt} \in \{0, 1\} \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 0, h_m, h_m+1, \ldots, T \tag{13}
\]

\[
B_{ct} \in \{0, 1\} \quad \text{for } c \in C, \ t = 1, 2, \ldots, T \tag{14}
\]

where \( X_{mt} \) is a binary decision variable whose value is 1 if management unit \( m \) is to be harvested in period \( t \) for \( t = h_m, h_m+1, \ldots, T \); when \( t = 0 \), the value of the binary
variable is 1 if management unit $m$ is not harvested at all during the planning horizon (i.e., $X_{m0}$ represents the “do-nothing” alternative for management unit $m$). $X_{mj}$ is the same thing as $X_{mt}$ but $j$ is used to indicate the period rather than $t$;

$h_m$ = the first period in which management unit $m$ is old enough to be harvested;

$\lambda$ = the minimum area of mature forest habitat patch over all periods;

$M$ = the number of management units in the forest;

$T$ = the number of periods in the planning horizon;

$c_{mt}$ = the discounted net revenue per hectare if management unit $m$ is harvested in period $t$, plus the discounted residual forest value based on the projected state of the stand at the end of the planning horizon;

$A_m$ = the area of management unit $m$ in hectares;

$v_{mt}$ = the volume of sawtimber in m$^3$/hectare harvested from management unit $m$ if it is harvested in period $t$;

$M_{ht}$ = the set of management units that are old enough to be harvested in period $t$;

$H_t$ = a continuous variable indicating the total volume of sawtimber in m$^3$ harvested in period $t$;
\( b_{lt} \) = a lower bound on decreases in the harvest level between periods \( t \) and \( t+1 \)

(where, for example, \( b_{lt} = 1 \) requires non-declining harvests, or \( b_{lt} = 0.9 \) would allow a decrease of up to 10%);

\( b_{ht} \) = an upper bound on increases in the harvest level between periods \( t \) and \( t+1 \)

(where, for example, \( b_{ht} = 1 \) allows no increase in the harvest level, or \( b_{ht} = 1.1 \) would allow an increase of up to 10%);

\( P \) = the set of all paths, or groups of contiguous management units, whose combined area is just above the maximum harvest opening size (the term “path,” as used in this paper, is defined in the following discussion);

\( P_i \) = the set of management units in path \( i \);

\( n_{P_i} \) = the number of management units in path \( i \);

\( h_i \) = the first period in which the youngest management unit in path \( i \) is old enough to be harvested;

\( O_{mt} \) = a binary variable whose value is 1 if management unit \( m \) meets the minimum age requirement for mature patches in period \( t \), i.e., the management unit is old enough to be part of a mature patch;
\( J_{mt} \) = the set of all prescriptions under which management unit \( m \) meets the minimum age requirement for mature patches in period \( t \);

\( C \) = the set of all clusters, or groups of contiguous management units whose combined area is just above the minimum large, mature patch size (the term “cluster,” as used in this paper, is defined in the following discussion);

\( S_c \) = the set of management units that compose cluster \( c \);

\( n_c \) = the number of management units in cluster \( c \);

\( B_{ct} \) = a binary variable whose value is 1 if all of the stands in cluster \( c \) meet the minimum age requirement for mature patches in period \( t \), i.e., the cluster is part of a mature patch;

\( BO_{mt} \) = a binary variable whose value is 1 if management unit \( m \) is part of a cluster that meets the minimum age requirement for large mature patches, i.e., the management unit is part of a patch that is big enough and old enough to constitute a large, mature patch;

\( C_m \) = the set of all clusters that contain management unit \( m \);
\( Age_{m_t}^T = \) the age of management unit \( m \) at the end of the planning horizon if it is harvested in period \( t \); and

\( \overline{Age}^T = \) the target average age of the forest at the end of the planning horizon.

Equation (1) specifies the first objective function of the problem, namely to maximize the discounted net revenue from the forest during the planning horizon, plus the discounted residual value of the forest. For age classes up to the optimal rotation, residual forest values are equal to the present value of the timber management costs and revenues on the management unit, assuming that it will be harvested at the optimal economic rotation, plus the present value of the land expectation value (LEV) representing future rotations. The LEV is the present value, per unit area, of the projected costs and revenues from an infinite series of identical even-aged forest rotations, starting initially from bare land. For age classes beyond the optimal economic rotation, residual forest values are equal to the liquidation value – i.e., the value of immediately harvesting the timber, plus the LEV for future rotations. Equation (2) maximizes the minimum amount of total area in large, mature forest patches over the time periods in the planning horizon.
Constraint set (3) consists of logical constraints that allow only one prescription to be assigned to a management unit, including a do-nothing prescription. Harvest variables \( X_{mt} \) are only created for periods where the stand is old enough to be harvested. Constraint set (4) consists of harvest accounting constraints that assign the harvest volume for each period to the harvest variables \( H_t \). Constraint sets (5) and (6) are flow constraints that restrict the amount by which the harvest level is allowed to change between periods. In the example below, harvests were allowed to increase by up to 15% from one period to the next or to decrease by up to 3%.

Constraint set (7) consists of adjacency constraints generated with the Path Algorithm (McDill et al. 2002). These constraints limit the maximum size of a harvest opening, often necessary for legal or policy reasons, by prohibiting the concurrent harvest of any contiguous set of management units whose combined area just exceeds the maximum harvest opening size. The exclusion period imposed by these constraints equals one planning period, but the constraints can be modified easily to impose longer exclusion periods in integer multiples of the planning period. A “path” is defined for the purposes of the algorithm as a group of contiguous management units whose combined area just exceeds the maximum harvest opening size. These paths are enumerated with a recursive
algorithm described in McDill et al. (2002). A constraint is written for each path to prevent
the concurrent harvest of all of the management units in that path, since this would violate
the maximum harvest opening size. This is done for each period in which it is actually
possible to harvest all of the management units in a path. (In the initial periods of the
planning horizon, some of the management units in a path may not be mature enough to be
harvested.)

Constraint sets (8)–(11) are the mature patch size constraints. Constraint set (8)
determines whether or not management units meet the minimum age requirement for
mature patches. These constraints sum over all of the prescription variables for a
management unit under which the unit would meet the age requirement for mature patches
in a given period. If any of these prescriptions have a value of 1, then $O_{mt}$ may also equal
1, indicating that the management unit will be “old enough” in that period. One of these
constraints is written for each management unit in each period.

Constraint set (9) determines whether or not a cluster of management units meets
the minimum age requirement for mature patches. Clusters are defined here as groups of
contiguous management units whose combined area just exceeds the minimum mature
patch size requirement. All possible clusters are enumerated using a recursive algorithm
described in Rebain and McDill (2003a). A cluster meets the age requirement for mature patches in period $t$ if all of the management units that compose that cluster meet the age requirement, as indicated by the $O_{mt}$ variables for the management units in that cluster. If cluster $c$ meets the age requirement in period $t$, then $B_{ct}$ is allowed to take a value of 1. These constraints are written for each cluster in each period.

Constraint set (10) determines whether or not individual management units are part of a cluster that meets the minimum age requirement, i.e., whether a management unit is part of patch that is big enough and old enough. Since the clusters overlap, this constraint set is necessary to properly account for the total area of large, mature patch habitat. These constraints say that a management unit is part of a patch that meets the minimum age and size requirement for large, mature patches in period $t$ ($BO_{mt} = 1$) if at least one of the clusters it is a member of meets the age requirement in that period. Constraint set (11) specifies that the total mature patch area for each period must be larger than $\lambda$ in all periods. Thus, $\lambda$ cannot be larger than the area of large, mature forest patch habitat in any period. Equations (2) and (11) work together to capture the minimum amount of total area in the large, mature forest patches over all the time periods (the value of the variable $\lambda$) and maximize it.
Constraint (12) is an ending age constraint. It requires the average age of the forest at the end of the planning horizon to be at least $\overline{Age}^T$ years, preventing the model from over-harvesting the forest. In the example below, the minimum average ending age was set at 40 years, or $\frac{1}{2}$ the optimal economic rotation. Constraint sets (13) and (14) identify the stand prescription and mature patch size variables as binary.

**Methods for Identifying The Efficient Frontier of the Bi-Criteria Model**

Several approaches have been developed to generate the efficient solution set for discrete multi-criteria optimization problems. This section briefly describes the basic methods and any variations from the original algorithms used in this research. The methods are described here primarily from the perspective of the objective space.

Whenever either the units or the scale of the values of the objectives are different, the criteria values must be normalized if a weighted objective function is used. In this research the ‘best value’ normalization approach was used, where the weight coefficients are divided by the appropriate elements of the ideal solution. The ideal solution is a vector whose elements are defined by the optimal attainment of the respective objective without regard to any of the other objectives (Figure 3-1). For example, the first element of the
The ideal solution vector for the example problem in this research is obtained by maximizing the net present value without regard to the minimum area of mature habitat; the second element is obtained by maximizing the minimum area of mature habitat without regard to the net present value. Clearly, the ideal solution is not attainable if the criteria conflict with one another. The ideal solution is identified and the criteria values are normalized in the initialization phase of each of the algorithms discussed below.

(1) The Weighted Objective Function Method (Pₗ)

Multiple-objective programming models, where the objective function is a weighted combination of multiple goals, have been applied to many forest and wildlife management planning problems (e.g., Roise et al. 1990, Hof and Joyce 1993, Snyder and Revelle 1997a, and Williams 1998). As the name implies, the weighted objective function method assigns weights to each of the objectives and combines them into a single scalar objective function. One way to determine a set of efficient solutions while maximizing the weighted objectives is to utilize the scalar maximum problem, known as the Pₗ problem, as proposed by Geoffrion (1968):

\[
Pₗ = \text{Max} \left\{ \sum_{i=1}^{n} \lambda_i f_i(x): \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \geq 0, \ x \in X \right\}
\]  

(15)
Where (15) maximizes the sum of the $P$ objective functions $f_i(x)$, weighted by scalars $\lambda_i \geq 0$, where the sum of the weights is 1, and the values of $x$ satisfy the constraints of the problem as defined by feasible set $X$. As mentioned above, since the scales and/or the units of the objectives are typically different, the weights have to be normalized. Assigning all combinations of weights to the objective functions guarantees the identification of each efficient point provided the following conditions are met:

1. **Theorem 1**: Let $\lambda_i > 0$ (i=1,...,$P$) be fixed. If $x^0$ is optimal for $P_\lambda$, then $x^0$ is an (properly\(^3\)) efficient solution (Geoffrion 1968).

2. **Theorem 2**: Let $X$ be a convex set, and let the $f_i$ be concave on $X$. Then $x^0$ is properly efficient if and only if $x^0$ is optimal in $P_\lambda$ for some $\lambda$ with strictly positive components (Geoffrion 1968).

Since the above spatially-explicit harvest scheduling problem involves discrete (binary) decision variables, the feasible set $X$ cannot be assumed to be convex for this

---

\(^3\) The concept of proper efficiency eliminates the situation where for some criterion the marginal gain in one objective can be made arbitrarily large relative to the marginal losses in each of the remaining criteria (Geoffrion 1968).
problem. Therefore, there is no guarantee that this method will generate all the efficient solutions. In fact, the Weighted Objective Function Method can only identify the supported strict Pareto optima. Nevertheless, the weighted objective function method can be used to create an initial set of solution alternatives. In an interactive approach, these alternatives may be presented to the DM, who can then specify the range within which further solution alternatives must be sought.

A modification of a well-known algorithm (c.f. Eswaran 1989) was used to decompose the weight space into sections (line segments in the bi-criteria case) that correspond to the same efficient solutions. In an ideal application of this method, a section can be eliminated from further exploration whenever its end points result in the same solution. However, the algorithm had to be modified slightly because large-scale problems cannot always be solved to exact optimality. The problems were solved with CPLEX 8.1, which uses a branch-and-cut algorithm to solve MIP problems. CPLEX was instructed to stop when the optimality gap – the percentage difference between the objective function value of the current best integer solution and the dual bound (McDill and Braze 2001) – reached 0.001%. While this is a very conservative stopping rule – the default value in CPLEX is 0.01% – there were cases where the solution found with one weight combination
dominated the solution found with an adjacent weight combination. By definition, the dominant solution would be a better solution for any weight combination, so the dominant solution was assumed to be the optimal solution for both weight combinations, and also for any weight combination in between them, and the line segment between the two weight combinations was not explored further. If the solutions corresponding to the end points of the line segment were different and neither dominated the other, a new weight combination was generated by calculating the mean of the two weight combinations at the end points. The new solution for the new weight combination was then compared with its neighboring solutions to determine whether the new subsections could be eliminated from further consideration. The algorithm was terminated when there were no sections left to decompose that were larger than a predetermined limit. This process is referred to as the decomposition algorithm; similar decomposition algorithms are used in some of the other methods described below.

(2) The $\varepsilon$-Constraining Method

This approach involves the following steps (Sadagopan et al. 1982). Step (1):

determine the ideal solution by solving for each objective without regard to the other. Call these optimal values Maximum Net Present Value (MNPV) and Maximum HABitat
(MHAB), respectively. **Step (2):** maximize NPV while constraining the minimum amount of large, mature habitat over all periods (HAB) to be larger than or equal to MHAB, and maximize HAB while constraining NPV to be larger than or equal to MNPV. This results in two efficient solutions that define the two ends of the efficient frontier. The remaining efficient solutions will be found within the rectangle defined by these two points. **Step (3):** choose a point on one of the criteria axes within the interval defined by the two points found in Step 2 (we chose the HAB axis). Call this value \( \overline{HAB} \). Maximize the other objective (NPV) on the feasible set, subject to an additional constraint that restricts HAB to be larger than or equal to \( \overline{HAB} \). Sadagopan et al.’s (1982) Theorem 1 states that this constraint will always be a binding constraint. Unfortunately, this solution (call it \( NPV_{\overline{HAB}} \)) might only be a weak Pareto optimal solution. Therefore, a fourth step is necessary to either confirm the efficiency of \( NPV_{\overline{HAB}} \) or find one solution that is efficient and dominates \( NPV_{\overline{HAB}} \). **Step (4):** Maximize HAB subject to the usual constraints, plus a constraint that requires NPV to be larger than or equal to \( NPV_{\overline{HAB}} \). Call this problem \( P_{\overline{HAB}} \). According to Sadagopan et al.’s (1982) Theorem 2, any solution that solves this problem is an efficient solution. The above two theorems enable us to find all efficient solutions by parametrically solving \( P_{\overline{HAB}} \) for different values of \( \overline{HAB} \) \( (0 < \overline{HAB} < MHAB) \).
Step 1. Obtain ideal solution:
(MNPV, MHAB)

Step 2. Max NPV st. HAB \geq MHAB
Max HAB st. NPV \geq MNPV
Call these efficient solutions:
EFS(1) = (NPV_{MHAB}, MHAB)
EFS(2) = (MNPV, HAB_{MNPV})
Set \( HAB = HAB_{MNPV}, k = 3. \)

Step 3. Max NPV st. HAB \geq \( \overline{HAB} + \delta. \)
Solution: \( NPV_{\overline{HAB}} \)

Step 4. Max HAB st. NPV \geq \( \overline{NPV} \)
Solution: \( HAB_{\overline{NPV}} \)
Set EFS(k) = (\( NPV_{\overline{HAB}}, HAB_{\overline{HAB}} \))

\( HAB = MHAB? \)

STOP

Figure 3-2. The \( \varepsilon \)-Constraining Algorithm

The algorithm used in this research, outlined in Figure 3-2, makes use of these two theorems by gradually proceeding from one end of the efficient frontier to the other. The first two steps are the same as those described by Sadagopan et al. (1982). Step 3 is to maximize NPV subject to a constraint that requires HAB to be larger than or equal to the HAB value from the previous solution plus a sufficiently small \( \delta. \) At the first iteration, this HAB value is equal to the objective function value of the solution that maximized HAB.
while constraining NPV to be larger than or equal to MNPV. The small $\delta$ is necessary to avoid the same solution that was obtained in the previous step. Of course, this value introduces the possibility that the algorithm will miss solutions that are within the interval defined by the arbitrary $\delta$. Step 4 is to maximize HAB subject to a constraint that restricts NPV to be larger than or equal to the NPV value obtained in step 3. The algorithm terminates when the HAB value reaches MHAB.

(3) The Decomposition Method based on the Tchebycheff-Metric

Eswaran et al. (1989) proposed a procedure to generate the entire efficient solution set for non-linear integer bi-criteria problems that uses the Tchebycheff metric and solves the following, so-called, $P_{\beta}$ problem for all parametric values of $\beta$:

\[
P_{\beta} = \min_{x \in \mathcal{X}} \left\{ \|f(x) - \bar{y}\|_{\beta} : \|f(x) - \bar{y}\|_{\beta} = \max_{i} \beta_i |f_i(x) - \bar{y}|, \sum_{i=1}^{p} \beta_i = 1, \ x \in \mathcal{X} \right\}
\]

(16)

where $\|f(x) - \bar{y}\|_{\beta} = \max_{i} \beta_i |f_i(x) - \bar{y}|$ is the weighted Tchebycheff-metric, $\bar{y}$ represents the ideal solution, $\beta_i$ is the weight parameter corresponding to objective $i$, and $f_i(x)$ is objective function $i$. 

86
All the solutions identified by the parametric decomposition of the \( \beta \)-space, which is analogous to parametric programming, are efficient solutions if the following sufficient condition, Bowman’s Theorem 4, is met: if an efficient set is uniformly dominant, then all the solutions to the \( P_\beta \) problem are efficient points (Bowman 1975). An efficient set is said to be uniformly dominant if, for every dominated point \( x^d \in X \), there exists an efficient point \( x^* \in X \) such that \( f_i(x^d) < f_i(x^*) \), for all \( i \) (Bowman 1975). In other words, Bowman’s Theorem 4 is upheld only when there are no weak Pareto optima (such as Point D in Figure 3-1). Since it is not possible in general to determine a priori whether weak Pareto optima exist for a given problem, we cannot conclude that the decomposition method based on the Tchebycheff metric will always identify strictly efficient solutions.

The same decomposition algorithm discussed in the Weighted Objective Function Method section above was used to decompose the weight space of the Tchebycheff-Metric. As discussed above, the algorithm applied here differs slightly from the one described by Eswaran et al. (1989) because we did not solve every problem to full optimality. Eswaran et al. (1989) assumed that all problems would be solved to optimality, so their algorithm eliminates sections of the weight space only when the end points result in the same solution. Our algorithm eliminates sections of the weight space either when the end points result in
the same solution or when the solution at one end point dominates the solution at the other end point. In the latter cases, the dominant solution was assumed to be the optimal solution at both end points and for all points in between.

(4) The Compromise Programming Method

Compromise Programming can also be used to generate efficient solutions for certain multi-criteria problems (Yu et al. 1975). This approach is based on the parametric minimization of the weighted $L_p$ metric, which measures the distance of solutions from the ideal solution with respect to each objective.

\[
P_{\text{comp.}} = \left\{ \min L_p = \left[ \sum_{i=1}^{K} \lambda_i^p \left[ y_i - f_i(x) \right] \right]^{\frac{1}{p}} : \sum_{i=1}^{K} \lambda_i = 1, \lambda_i \geq 0, x \in X \right\}, \tag{17}
\]

where the $\lambda_i$ are weights arbitrarily assigned to each objective (they could be set the same for each objective), $p = 1, 2, \ldots, \infty$ is a parameter, and $y_i$ is the ideal solution for objective $i$.

The $L_p$ metric is a very general approach, where each specific $p$-value results in a different method. For example, setting $p = \infty$ is equivalent to using the Tchebycheff-metric. The major drawback of this method, however, is that its formulation results in non-
linear problems for p-values other than 1 or $\infty$. Considering the complexity and size of the harvest scheduling problem, a non-linear approach is out of the question.

(5) Hybrid Methods

A number of hybrid methods have been described in the operations research literature that, by combining some of the above basic approaches, efficiently utilize the positive features of more than one method. For example Wendell and Lee (1977) combined the Weighted Objective Function Method with the $\varepsilon$-Constraining Method (Wendell and Lee 1977). They fixed the weight coefficients, $\lambda_i$ and parametrically solved the problem below for each $\varepsilon_i$. The advantage of the method is that the weight coefficients do not have to be altered.

$$\text{Max } P_{\text{hybrid}} = \text{Max} \left\{ \sum_{i=1}^{P} \lambda_i f_i(x) : f_i(x) \geq \varepsilon_i, \sum_{i=1}^{P} \lambda_i = 1, \lambda_i \geq 0, x \in X \right\} \quad (18)$$

for all $i = 1,\ldots,P$, where $f_i(x)$ is objective function $i$. Both of the algorithms below can be thought of as special cases of the hybrid method introduced by Wendell and Lee (1977). The Alpha-Delta Method was developed by the authors, and the Triangles Method by Chalmet et al. (1986).
(6) The Alpha-Delta Method

This approach takes advantage of the fact that if we assign an incomparably larger weight to one objective than to the other, strong Pareto optima can be identified consecutively along the efficient frontier using a procedure similar to the ε-Constraining method. Figure 3-3 illustrates this process. The initialization phase is the same as in the Decomposition Method based on the Tchebycheff-metric: calculate the ideal solution and then the two end points of the efficient frontier, (EFS(1) and EFS(2)), as in Figure 3-2. A very large weight is then assigned to one objective and a minimal weight to the other. In Figure 3-3, PQ demonstrates such an allocation of weights. From here on, the objective with the higher weight (NPV in our case) is maximized at each step subject to a constraint that requires the achievement value of the other objective (HAB) to be greater than or equal to the achievement value obtained by the previous step plus a sufficiently small δ. At the first iteration, this achievement value is equal to the objective function value of the solution that maximized the HAB objective while constraining NPV to be larger than or equal to MNPV (MNPV is the first element of the ideal solution vector). The small δ ensures that a new solution will be found. For example, using the weighted objective function, PQ in Figure 3-3a, Point A would be picked up repeatedly if the lower bound on HAB were not augmented by δ (AHab + δ). Instead, Point B will be found next (Figure 3-3a). The next
iteration is implemented using the new lower bound of \((BH_{ab} + \delta)\), where the \(BH_{ab}\) value was obtained in the previous step.

The parameter \(\delta\) has to be set to a small value to minimize the probability that efficient points will be missed. In Figure 3-3b, for example, Point C would be missed if \(\delta\) were not reduced to least half of its current value. Similarly, the parameter \(\alpha\) (the slope of the weighted objective function) has to be small to minimize the probability that an efficient point will be missed. The algorithm terminates when the achievement value of the habitat reaches its upper bound (MHAB in our case). The advantage of this algorithm is
that the new solution at each step will always neighbor the previous one along the efficient frontier if sufficiently small $\alpha$ and $\delta$ are used, and, while the $\varepsilon$-Constraining Method finds each new solution in two steps, this approach will do it in one.

(7) The Triangles Method

This algorithm, developed by Chalmet et al. (1986), seeks Pareto optimal solutions between two adjacent, efficient points that have already been identified (e.g., solutions A and B in Figure 3-4). The weight coefficients on the objective functions are fixed and arbitrary. We used equal weights for both objectives in this research. At each step, the search space (the grey area in Figure 3-4) is confined by two constraints. These constraints are gained by adding a small $\delta_1$ and $\delta_2$ to the lower achievements on the two objectives at the two adjacent solutions. In Figure 3-4, for example, the section of the efficient frontier between Point A and Point B is explored (the grey area in Figure 3-4a). A section between two adjacent efficient points will be eliminated from further investigation if no feasible solution is found there (such as in Figure 3-4b) or, alternatively, if the difference in one of the objective values between the two solutions is smaller than a predetermined limit. The algorithm terminates when there are no sections left to explore. Again, $\delta_1$ and $\delta_2$ have to be
small to minimize the possibility that the algorithm will miss an efficient solution. As an example, in Figure 3-4b, Point C would be missed if the value of $\delta_1$ were not reduced.

A case study

In order to illustrate and test the performance of the various algorithms for generating the efficient frontier, an example hypothetical forest was created. This forest consisted of 50 stands and could be considered slightly over-mature, since approximately 40% of the area is between 60-100 years old and the optimal rotation is 80 years. The average stand size was 18 ha, and the total forest area was 900 ha. A 60-year planning horizon was considered, composed of three 20-yr periods. The four possible prescriptions
for a given stand were: cut the management unit in period 1, period 2, or period 3, or do not cut it at all. The minimum rotation age was 60 years. A maximum harvest opening size of 40 ha was imposed, and adjacent stands were allowed to be harvested concurrently as long as they did not violate this maximum opening size. All management units are smaller than the maximum harvest opening size. The wildlife species under consideration is assumed to need habitat patches that are at least 50 ha in size and at least 60 years old. Since the minimum habitat patch size is greater than the maximum harvest opening size, these patches must be composed of more than one management unit.

We implemented the algorithms described in the Methods section using CPLEX 8.1 (ILOG CPLEX 2002) on a Dual-AMD Athlon™ MP 2400+ (2.00 GHz) computer with 2.0 GB RAM. Programs to automate the algorithms were written in Microsoft Visual Basic 6 using the ILOG CPLEX Callable Libraries. The relative MIP gap tolerance parameter (optimality gap) was set to 0.00001 (0.001%), and the MIP variable selection strategy parameter was set to ‘3’ (i.e., strong branching).

The experiment addressed the following questions: (1) How many of the efficient solutions can each algorithm identify? (2) How long does each algorithm take to identify all of the solutions that are found? and (3) How good are these solutions in terms of
optimality? The third question refers to the fact that even though the optimality gap was set to 0.001% for each algorithm some methods might consistently generate solutions that are better than the ones generated by other methods but still within this range.

![Figure 3-5. The efficient frontier of a bi-criteria harvest scheduling problem](image)

**Results and Discussion**

*Figure 3-5* shows the efficient frontier generated by the various methods. The Weighted Objective Function Method identified only six efficient solutions (points A, E, C, F, G, D on *Figure 3-5*). The $\varepsilon$-Constraining, the Alpha-Delta, and the Triangles Methods
all found the highest number of efficient solutions (36). In terms of solution times, however, Alpha-Delta Method was considerably faster than the others (6.27 hrs), followed by the Triangles Method (17.13 hrs), and then the ε-Constraining Method (58.75 hrs). The Tchebycheff Decomposition Method found 34 solutions in 36.83 hours.

Table 3-1. The set of efficient solutions

<table>
<thead>
<tr>
<th>Efficient solutions</th>
<th>Efficient solutions missed</th>
<th>No.</th>
<th>NPV ($)</th>
<th>Habitat (ha)</th>
<th>by the respective methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>2,445,084</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>2</td>
<td>2,427,424</td>
<td>64.3817</td>
<td>W</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3</td>
<td>2,423,119</td>
<td>64.9906</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>4</td>
<td>2,410,088</td>
<td>66.3003</td>
<td>W</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>5</td>
<td>2,409,663</td>
<td>66.7425</td>
<td>W</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>6</td>
<td>2,397,375</td>
<td>79.0436</td>
<td>W</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>7</td>
<td>2,395,672</td>
<td>85.2725</td>
<td>W</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>8</td>
<td>2,373,040</td>
<td>101.2655</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>9</td>
<td>2,372,633</td>
<td>119.0577</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>10</td>
<td>2,336,542</td>
<td>121.2323</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>11</td>
<td>2,327,001</td>
<td>121.3514</td>
<td>W</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>12</td>
<td>2,318,702</td>
<td>122.6209</td>
<td>W</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>13</td>
<td>2,317,408</td>
<td>123.2003</td>
<td>W</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>14</td>
<td>2,317,016</td>
<td>126.3608</td>
<td>W</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>15</td>
<td>2,317,008</td>
<td>126.5215</td>
<td>W</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>16</td>
<td>2,316,923</td>
<td>128.2639</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>17</td>
<td>2,316,392</td>
<td>129.7142</td>
<td>W</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>18</td>
<td>2,312,040</td>
<td>130.0187</td>
<td>W</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td>19</td>
<td>2,311,517</td>
<td>134.9016</td>
<td>W</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>20</td>
<td>2,311,161</td>
<td>136.1711</td>
<td>W</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>21</td>
<td>2,308,774</td>
<td>136.8799</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>22</td>
<td>2,276,022</td>
<td>137.0762</td>
<td>W</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>23</td>
<td>2,273,635</td>
<td>137.785</td>
<td>W</td>
</tr>
<tr>
<td>24</td>
<td></td>
<td>24</td>
<td>2,268,411</td>
<td>138.2501</td>
<td>W</td>
</tr>
<tr>
<td>25</td>
<td></td>
<td>25</td>
<td>2,268,166</td>
<td>138.4648</td>
<td>W</td>
</tr>
<tr>
<td>26</td>
<td></td>
<td>26</td>
<td>2,267,415</td>
<td>139.0545</td>
<td>W</td>
</tr>
<tr>
<td>27</td>
<td></td>
<td>27</td>
<td>2,263,109</td>
<td>139.1736</td>
<td>W</td>
</tr>
<tr>
<td>28</td>
<td></td>
<td>28</td>
<td>2,255,576</td>
<td>140.4431</td>
<td>W</td>
</tr>
<tr>
<td>29</td>
<td></td>
<td>29</td>
<td>2,252,026</td>
<td>143.2915</td>
<td>W</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>30</td>
<td>2,250,014</td>
<td>152.015</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>31</td>
<td>2,246,204</td>
<td>152.7238</td>
<td>W</td>
</tr>
<tr>
<td>32</td>
<td></td>
<td>32</td>
<td>2,238,791</td>
<td>153.9933</td>
<td>W</td>
</tr>
<tr>
<td>33</td>
<td></td>
<td>33</td>
<td>2,205,934</td>
<td>154.8984</td>
<td>W</td>
</tr>
<tr>
<td>34</td>
<td></td>
<td>34</td>
<td>2,194,305</td>
<td>156.287</td>
<td>W</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td>35</td>
<td>2,174,106</td>
<td>163.5352</td>
<td>W</td>
</tr>
<tr>
<td>36</td>
<td></td>
<td>36</td>
<td>2,170,416</td>
<td>169.8372</td>
<td></td>
</tr>
</tbody>
</table>

Notes: The ‘W’s and ‘Ts’s stand for those efficient points that were missed by the Weighted Objective Function and the Tchebyseff methods respectively.
Table 3-2. Differences in solution optimality between the various methods

<table>
<thead>
<tr>
<th>ε-Constraining &amp; Triangles</th>
<th>Alpha-Delta</th>
<th>Difference</th>
<th>Tschebyseff</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>No.</td>
<td>NPV ($)</td>
<td>Habitat (ha)</td>
<td>NPV ($)</td>
<td>Habitat (ha)</td>
</tr>
<tr>
<td>2</td>
<td>2,427,424</td>
<td>64.3817</td>
<td>2,426,808</td>
<td>64.3817</td>
</tr>
<tr>
<td>3</td>
<td>2,423,119</td>
<td>64.9906</td>
<td>2,423,028</td>
<td>64.9906</td>
</tr>
<tr>
<td>4</td>
<td>2,410,088</td>
<td>66.3003</td>
<td>2,409,557</td>
<td>66.3003</td>
</tr>
<tr>
<td>5</td>
<td>2,409,663</td>
<td>66.7425</td>
<td>2,409,088</td>
<td>66.7425</td>
</tr>
<tr>
<td>36</td>
<td>2,170,416</td>
<td>169,6372</td>
<td>2,170,320</td>
<td>169,6372</td>
</tr>
</tbody>
</table>

Table 3-1 summarizes the set of efficient solutions. In terms of optimality, the ε-Constraining and the Triangles Methods performed the best. The Alpha-Delta Method provided the same solutions as those generated by ε-Constraining or the Triangles Method in all but five cases. In those cases, the achievements of the NPV objective were slightly less (Table 3-2). In 3 out of 34 cases, the Tchebycheff Decomposition Method resulted in lower NPV achievements than the ε-Constraining or the Triangles Methods. As the greatest difference in NPV was only 0.0254%, these differences are probably not a significant concern. However, it is noteworthy that the differences consistently favor some algorithms over others.

Figure 3-6 shows the cumulative time required to obtain each solution for each method. The figure clearly shows that the Alpha-Delta Method dominates the others in terms of solution time, finding all 36 efficient solutions in a little more than six hours. The figure also shows that the Alpha-Delta and the Triangles Methods found all 36 efficient
solutions well before the $\varepsilon$-Constraining found 5. Although the Weighted Objective Functions Method identified only six of the efficient points, these six points were found relatively quickly and this “filtered” set of alternatives might be useful for interactive methods involving the DM or to find a good, distributed set of alternative solutions if solution time is a constraint. There is no guarantee, however, that the set of solutions found with this method will be evenly distributed along the efficient frontier, as they are in this case (see Figure 3-5).

Figure 3-6. Cumulative solution times for each method
While none of the methods can guarantee that they will find all of the efficient solutions to a problem, three of the five methods that were tested found essentially the same set of 36 efficient solutions. The Tchebycheff Decomposition algorithm does not guarantee the identification of the complete set of efficient alternatives unless the uniformly dominant property of the feasible set of the harvest scheduling problem holds. This method failed to find two of the 36 efficient solutions found by the other methods. In general, it is not likely that the uniformly dominant property will hold for problems like the example problem used here. For example, there generally are a very large number of feasible solutions with a given minimum area of mature patch habitat in one period, but with varying net present values. It is hard to predict how many efficient solutions this method would actually find for any given problem. In contrast, by adjusting the parameters of the $\varepsilon$-Constraining (parameter $\delta$), Alpha-Delta (parameters $\delta$ and $\alpha$) or Triangles (parameters $\delta_1$ and $\delta_2$) Methods, one can reduce the probability of missing any of the solutions with minimal additional computational cost. It is likely that, for similar parameter settings, the chance that the $\varepsilon$-Constraining Method will miss an efficient solution is lower than the chance of missing a solution with Alpha-Delta or Triangles Methods as the former has only one parameter, and hence only one area, that controls the size of the area where missed
solutions might exist. In addition to not finding all of the efficient solutions, the time required by the Tchebycheff Method to find the 34 efficient solutions that it found was substantially longer than the time required by either the Alpha-Delta or the Triangles Methods to find 36 efficient solutions.

The potential of the methods discussed here to generalize to the tri-criteria case is difficult to assess without actually applying them to a specific case. However, a few observations can be made at this time with regard to this issue. First, the Weighted Objective Functions Method is relatively easy to generalize to the n-dimensional case, but the results here suggest that this method can miss many of the efficient solutions. Furthermore, the parametric decomposition of a 3-dimensional weight-space, which is central to the Tchebycheff approach, is readily available from the OR literature (c.f. Ravindran 2003). However, as this procedure makes use of the traditional sensitivity analysis of the linear programming theory, it is not straightforward to apply this process in the case of spatially-explicit harvest scheduling models with discrete variables. The reduced costs with respect to each objective are needed to calculate the ranges of weights within which a given efficient solution is optimal. These reduced costs, however, are not available for discrete variables. Additionally, as we have seen, this method may miss some
efficient solutions and is not particularly efficient with regard to solution time.

Generalizing the other three algorithms to deal with three objectives appears to be quite complicated but not infeasible. Each approach would require either adding a potentially large number of constraints to the problem, or creating several sub-problems at each iteration. For the Alpha-Delta and the $\varepsilon$-Constraining Methods, the set of constraints that one might add to the formulation at each step would form a non-convex feasible region in the objective space. By introducing a set of binary variables, this region can easily be described within one formulation. An alternative approach would be to create several sub-problems. Generalizing the $\varepsilon$-Constraining approach to more than two objectives would be particularly expensive computationally since, at each step, an efficient solution can only be obtained after solving three sub-problems for the tri-criteria case ($n$ sub-problems for the $n$-criteria case). This is the only way, however, to ensure that the final solution is not dominated. This problem is avoided by using a weighted objective function with non-zero, fixed weights in the Alpha-Delta or Triangles algorithms. In our experience, the Alpha-Delta Method has the further advantage, over the Triangles Method, of being very simple to translate into computer code.
Conclusions

The multi-criteria optimization techniques discussed in this paper provide useful alternatives to goal programming or other multi-criteria approaches when the decision maker does not have a prior understanding of the potentials for and trade-offs between the conflicting objectives and therefore cannot readily specify preferences or a list of targets for the objectives. This situation occurs frequently in forest planning. Target values or preferences for criteria that describe wildlife habitat goals, such as the overall area to maintain in mature forest patches or the amount of edges within a given landscape, are often hard to specify a priori. By providing exact information on the nature of the trade-offs between such conflicting criteria, the methods discussed above would help the DM select the best compromise solution and give him or her more insight into the problem. We believe that this is the primary value of the “frontier” methods described here. In those situations where the DM is confident about what the targets should be, these computationally expensive methods may not be appropriate.

The following conclusions are suggested by the theoretical discussion and the analysis of the test problem in this paper: (1) If a complete set of efficient solutions is desired, the discrete nature of the harvest scheduling problem rules out the weighted
objective function method and compromise programming as useful approaches because many efficient solutions may be missed or non-linear programming would be needed; (2) In the bi-criteria case, the ε-Constraining, the Tchebycheff Decomposition, the Alpha-Delta, and the Triangles Methods are all capable of identifying a very good set of solutions; (3) The Alpha-Delta, and the Triangles Methods performed the best in terms of solution times for the test problem; (4) There were infrequent and minor differences in how the different algorithms performed in terms of solution optimality, but when differences occurred, the ε-Constraining and the Triangles Method consistently performed better than the other methods; (5) In our experience, the Alpha-Delta Method is the easiest to translate into computer code; (6) Although each of the methods can be generalized to the tri-criteria case, the Alpha-Delta Method appears to generalize the most easily.

Rigorous additional experimentation would be needed to determine whether these results would apply to a wide range of forest planning problems of various scales and various structures. The primary avenue of future research, however, points to the development of algorithms that would efficiently tackle the general, n-criteria case for discrete formulations such as the spatially-explicit harvest scheduling problems. As multiple-use forest management becomes more important for society, multi-objective
optimization techniques, such as the “frontier” methods discussed here, will probably receive more attention in the future. Furthermore, their potential will rapidly expand as the performance of both optimization software and computer hardware improves. It is also likely that the interactive utilization of the “frontier” methods has a lot of potential for multi-criteria forest planning, as the involvement of various stakeholders in the decision making process will become increasingly important.
Abstract:

Designing harvest scheduling schemes that provide the public with an optimal bundle of timber and non-timber benefits is a major challenge of the forest planning community today. Spatially-explicit harvest scheduling models capable of promoting or enforcing the development of mature forest habitat patches of a minimum size had been proposed in the past. The linear integer programming formulations introduced in this paper ensure that the total perimeter of these mature forest patches is as small as possible. This research assesses the extent to which the proposed formulations can promote compact (low perimeter-area ratio) mature forest patches, and at what cost. The impact of these formulations on other spatial attributes, such as the number of the patches within the landscape was also analyzed. The objectives were tested with a 50-stand hypothetical forest. The results suggest that the proposed formulation promotes compactness and tends to result in fewer, larger patches at a relatively low cost.
**Keywords:** spatially-explicit harvest scheduling, interior habitat, minimum boundary, adjacency constraints, area restriction model

**Introduction**

Forest resources management usually involves making decisions while addressing multiple objectives. This is particularly true for public forests. Society expects more from its forest resources than merely timber production. Non-timber benefits such as wildlife habitat, water quality, recreation, aesthetics and spiritual values are increasingly important to people. On the other hand, excluding timber production as one of the objectives of forest management in public forests tends to export the burden on forest resources to other suppliers where sustainable forest stewardship might be less likely to occur (Thomas 2000).

The Multiple-Use Sustained-Yield Act (1960) recognizes the multiple-objective nature of forest management by requiring the national forests of the United States to be managed for the multiple uses of water, timber, wildlife, fish, recreation, and range (Fedkiw 1997).

Unfortunately, timber and non-timber objectives often conflict with each other. Thus, designing cost-efficient management schemes that provide the public with an optimal combination of timber and non-timber benefits is a major challenge for forest planners.
Tools that can provide information about the exact nature of these conflicts can be very useful. Being able to demonstrate what it costs to pursue a non-timber objective, for example, or how much mature forest habitat would have to be forgone in order to achieve a timber production goal, can have an enormous impact on decision making. Thus, being able to visualize the tradeoffs between timber and non-timber objectives can help decision makers (DM) better understand the forest management problem and be more informed when selecting the best compromise management alternative.

Spatially-explicit harvest scheduling models can be excellent multiple-objective decision support tools for forest management (Tóth et al. in press). Harvest scheduling models have been used in the United States for 30 years to help identify the most profitable management alternatives for a given forest. These models were initially formulated as linear programs (LP) (e.g., Johnson and Scheurman 1977). The output of these LPs indicate how many acres of a certain analysis area – an aggregate forest area with similar silvicultural characteristics – should be treated with a given prescription in a given time period in order to maximize profit, timber output, or some other objective. Sustainability concerns, such as a target age class distribution or a smooth flow of timber production over time, have typically been represented as constraints. The ability of these LP models to
address spatial constraints was limited, however. In general, they could indicate when and how much of a forest resource type should be treated but they did not specifically indicate where. Furthermore, non-timber objectives or constraints that are fundamentally spatial in nature, such as conservation of mature forest habitat meeting a minimum size requirement or habitat connectivity were either very cumbersome or impossible to model.

Recently, spatially-explicit mixed-integer harvest scheduling models have been introduced that can address, to some extent, these shortcomings. In these models, the decision whether or not a forest management unit should be harvested in a given time period is represented by 0-1 variables. The introduction of these 0-1 variables provides more flexibility in incorporating non-timber objectives in the modeling framework. These models have been studied extensively in the context of applying adjacency constraints, whose function is to limit the size of clear cuts. These constraints have been promoted as contributing to many non-timber objectives (e.g. Thompson et al. 1973, Jones et al. 1991, Murray and Church 1996a, 1996b, Snyder and ReVelle 1996a, 1996b, 1997a, 1997b, Carter et al. 1997, Murray 1999). However, adjacency constraints tend to disperse harvests across the landscape, which tends to be counterproductive if the management goals include preserving or fostering the development of large patches of mature forest (Harris 1984).
Franklin and Forman 1987). To mitigate this ‘dispersion effect’, Rebain and McDill (2003b) proposed a model formulation capable of promoting or enforcing the development of large mature forest patches – patches that consist of forest stands that are older than a certain age and larger than a certain size.

Rebain and McDill’s (2003b) model requires a certain amount of mature forest habitat to be preserved in patches meeting minimum age and size requirements at any given time (planning period) within the planning horizon. However, the mature forest patches created by their model may change from one period to the next. In other words, the patches are allowed to float dynamically on the landscape. This way, areas of mature forest habitat will exist in some location on the landscape at any given point in time, but species dependent on mature forest habitat might need to find new, suitable habitats when their original habitat is lost (i.e., harvested). Although this is not a perfect solution for every species, especially for those with limited capabilities for dispersal, this ecosystem management strategy might still be a good option in landscapes that lack reserves (Spies et. al. 2000).

Another limitation of the Rebain and McDill (2003b) model is that it only requires mature forest patches to meet age and size requirements, but the shape of these patches may
also be important. Shape is an important parameter in habitat conservation because mature forest patches of the same size but different shapes can provide very different habitats for wildlife. For example, due to the edge-effect, large and mature forest patches do not necessarily provide interior habitat if their shape is too elongated (Gustafson and Crow 1998). Patches with complex shapes have proportionately more edge habitat than those of similar area but roughly circular shape (Franklin and Forman 1987). In general, managed forest ecosystems have more edge habitat than pristine ecosystems because harvesting creates a lot of edge.

Forest edges – defined as “abrupt transitions between two relatively homogeneous ecosystems, at least one of which is a forest” (Matlack et. al. 2000, pp.210) – generate unique habitats. These so-called ‘edge habitats’, if numerous, have a profound impact on the overall integrity of the forest ecosystem. On the negative side, they can reduce biodiversity in the forests (e.g. Gates and Gysel 1978, Yahner 1988, Paton 1994), attract predators and therefore might function as ‘ecological traps’ (Gates and Gysel 1978), and provide habitat that favors some invasive, often harmful plant species (Matlack et. al. 2000). On the other hand, edge benefits some native species, including some whose habitat is in decline; such as the golden-winged warbler (Vermivora chrysoptera) or the bobcat (Felis
rufus) in the northeastern United States (Matlack et. al. 2000). In landscapes where the negative impacts of edges outweigh their positive impacts, Matlack et. al. (2000) recommend designing harvest layouts that minimize the length of edges relative to interior habitat.

The proportion of edge habitat within a patch is determined by the shape of the patch. Patches of circular shape are the most compact. In particular, the perimeter-area ratio (PAR) of a patch is correlated to the proportion of edge habitat; the lower the PAR of a patch, the more compact it is and the less edge habitat it tends to contain. Baker and Cai (1992) argue that PAR as a metric might not be the best choice to account for the shapes of patches as its value changes with increasing size, even though the shape of the patch remains the same. We use PAR as the measure of desirable shape for the purposes of this study because from an ecological perspective larger habitat patches should receive higher scores for compactness than smaller ones with the same shape as they provide more interior habitat.

Two concerns, therefore, regarding Rebain and McDill’s (2003b) model are that (1) the temporal and spatial connectivity (or overlap) of the patches is not ensured, and (2) there is nothing in the model that helps the forest planner control the shape of the patches.
This study attempts to address the latter issue. Pursuing compactness (i.e., low perimeter-area ratio) can reduce the edge effect. Still, compactness is desirable only if the conservation of interior habitat is the goal. If the goal is to provide more edge habitat, or there is a concern that promoting compactness may conflict with the connectivity of patches, then generating patches with high PAR should be pursued.

In forest planning problems where the preservation of interior mature forest habitat is a management objective the most natural way of promoting compactness is to minimize the edge length of the mature forest patches while keeping their area above a certain limit. In the context of the present study, then, edge is defined as a transition between mature forest habitat patches and everything else.

Similar perimeter minimizing approaches have been followed by a number of studies in the optimal reserve selection literature. The optimal reserve selection problem, a variant of the set covering problem, seeks the minimum cost representation of a set of species within a network of candidate reserves or maximizes the species representation given budget constraints. Spatial attributes of the network such as the relative proximity, the connectedness, or the compactness of the reserves have been extensively modeled (e.g.,

Williams and Revelle (1996, 1998) incorporated the minimum boundary concern into their model by requiring that buffers of unit width must surround the core reserve areas. As cost minimization favors the selection of as few buffers as possible, the model indirectly promotes compactness and contiguity. Their assumption that the reserve parcels form a regular grid network, where the parcels are squares of unit acreage, made the integer programming (IP) formulation simple and tight. However, their approach is less attractive in the case of harvest scheduling, where the shapes of the cutting units are typically predefined and irregular, which would make it difficult to control the width of the buffer.

Önal and Briers (2003) introduced an IP formulation that minimizes the boundary of the reserve networks. Using three sets of binary decision variables, their model calculates the boundary length of the reserve by determining whether an edge is part of the boundary of the reserve network or is an interior segment. McDonnell et al. (2002) introduced a non-linear formulation of the minimum boundary criterion that can be applied to irregularly shaped patches. They account for compactness through a weighted objective function. However, determining the weight that should be assigned to this criterion is not trivial.
Furthermore, the non-linear nature of the formulation severely restricts the potential of this model to exactly optimize large-scale problems. Fischer and Church (2003) employed a similar approach but recognized that the perimeter term used by McDonnell et al. (2002) can easily be linearized. This is a significant improvement because the resulting linear structure allows powerful IP solvers to tackle larger problems.

Minimizing the boundary of mature forest patches in harvest scheduling models requires a slightly different approach. This is because, unlike reserves, mature forest patches in harvest scheduling models are allowed to dynamically float across the landscape over time. The added temporal dimension makes harvest scheduling models more complicated. It is not clear for instance, whether minimizing the total perimeter of the patches that evolve throughout the planning horizon results in lower PAR values (and, thus, in more compactness), or minimizing the maximum of the perimeters that correspond to the respective planning periods.

This research builds on the work of Rebain and McDill (2003a, b) and explores ways to identify management alternatives that would result in the development of large mature forest patches with minimum boundaries. Alternative formulations will be presented and compared in terms of compactness achieved and computational difficulty.
The cost of perimeter minimization will be evaluated based on how much net present value would have to be forgone in order to achieve minimum boundaries. In addition, the impact of the compactness formulation on the fragmentation of mature forest habitat will be examined by comparing the total number of patches that evolve over the planning horizon with and without perimeter minimization.

The primary value of the models presented in this paper is that they can provide quantitative information on the tradeoffs between timber production and mature forest habitat conservation. Suppose a DM who is responsible for endorsing the management plan for a 50,000 ha state forest would be willing to forgo $2,000,000 in timber revenues to ensure that at least one fifth of the total forest area is preserved in large, compact patches of mature forest habitat at any given planning period. Is this guideline feasible? Could considerably more habitat be maintained at a much lower cost? The models proposed in this paper might help answer these types of questions.

**Model Formulations**

This section describes two slightly different IP formulations that minimize the boundary of the mature forest patches that evolve over the planning horizon. One
minimizes the maximum amount of edge that surrounds these patches over the time periods in the planning horizon (MINIMAX approach). The other minimizes the total length of edges over the entire planning horizon (TOTALMIN approach). Boundary minimization is imbedded as an objective function in a tri-criteria spatially-explicit harvest scheduling problem. The other objective functions maximize the net present value (NPV) of the forest, and the minimum amount of total area in large, mature forest patches over the time periods in the planning horizon. The models include harvest flow constraints, maximum harvest opening size constraints, constraints that define the minimum area of large, mature patch habitat over time, and a minimum average ending age constraint. The model formulation of the mature forest patch criterion is a slightly modified version of the one presented in Rebain and McDill (2003a). Formulation of the maximum harvest area constraints is a generalization of the formulation presented in McDill et al. (2002).

The MINIMAX Model:

\[ Max \ Z = \sum_{m=1}^{M} A_m [c_{m0} X_{m0} + \sum_{t=h_e}^{T} c_{mt} X_{mt}] \]  \hspace{1cm} (1)

\[ Max \ \lambda \]  \hspace{1cm} (2)
Min \ \mu \\
\text{subject to:}

\begin{align*}
X_{m0} + \sum_{t=h_m}^T X_{mt} \leq 1 & \quad \text{for } m = 1, 2, \ldots M \quad (4) \\
\sum_{m=1}^M v_{mt} \cdot A_m \cdot X_{mt} - H_t = 0 & \quad \text{for } t = 1, 2, \ldots T \quad (5) \\
b_{ht} H_t - H_{t+1} \leq 0 & \quad \text{for } t = 1, 2, \ldots T-1 \quad (6) \\
-b_{ht} H_t + H_{t+1} \leq 0 & \quad \text{for } t = 1, 2, \ldots T-1 \quad (7) \\
\sum_{m \in M_t} X_{mt} \leq n_{tP} - 1 & \quad \text{for all } P_t \in P \text{ and } t = h_i, \ldots , T \quad (8) \\
\sum_{j \in J_m} X_{mj} - O_{mt} \geq 0 & \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots , T \quad (9) \\
\sum_{j \in J_m} X_{mj} - |J_{mt}| O_{mt} \leq 0 & \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots , T \quad (10) \\
\sum_{m \in S_c} O_{mt} - n_c B_{ct} \geq 0 & \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots , T \quad (11) \\
\sum_{m \in S_c} O_{mt} - B_{ct} \leq n_c - 1 & \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots , T \quad (12)
\end{align*}
\[
\sum_{c \in C_w} B_{ct} - BO_{mt} \geq 0 \quad \text{for } m = 1, 2, \ldots M, \text{and } t = 1, 2, \ldots, T \quad (13)
\]

\[
\sum_{c \in C_w} B_{ct} - |C_m| BO_{mt} \leq 0 \quad \text{for } m = 1, 2, \ldots M, \text{and } t = 1, 2, \ldots, T \quad (14)
\]

\[
\sum_{m=1}^{M} A_m BO_{mt} \geq \lambda \quad \text{for } t = 1, 2, \ldots T \quad (15)
\]

\[
\sum_{m=1}^{M} P_m BO_{mt} - 2 \sum_{pq=1}^{N} CB_{pq} \Omega_{pq}^t \leq \mu \quad \text{for } t = 1, 2, \ldots T \quad (16)
\]

\[
BO_{pt} + BO_{qt} - 2\Omega_{pq}^t \geq 0 \quad \text{for } t = 1, 2, \ldots T, pq = 1, 2, \ldots N \quad (17)
\]

\[
BO_{pt} + BO_{qt} - \Omega_{pq}^t \leq 1 \quad \text{for } t = 1, 2, \ldots T, pq = 1, 2, \ldots N \quad (18)
\]

\[
\sum_{m=1}^{M} A_m [(Age_{0t}^T - \overline{Age}^T)X_{0t} + \sum_{t=h_m}^{T} (Age_{mt}^T - \overline{Age}^T)X_{mt}] \geq 0 \quad (19)
\]

\[
X_{mt} \in \{0,1\} \quad \text{for } m = 1, 2, \ldots M, \text{and } t = 0, h_m, h_m + 1, \ldots, T \quad (20)
\]

\[
B_{ct} \in \{0,1\} \quad \text{for } c \in C, \ t = 1, 2, \ldots T \quad (21)
\]

\[
O_{mt}, BO_{mt}, BO_{pt}, BO_{qt} \in \{0,1\} \quad \text{for } m, p, q = 1, 2, \ldots M, \ t = 1, 2, \ldots T \quad (22)
\]

\[
\Omega_{pq} \in \{0,1\} \quad \text{for } pq \in N \quad (23)
\]
where \( X_{mt} \) = a binary decision variable whose value is 1 if management unit \( m \) is to be harvested in period \( t \) for \( t = h_m, h_{m+1}, \ldots, T \); when \( t = 0 \), the value of the binary variable is 1 if management unit \( m \) is not harvested at all during the planning horizon (i.e., \( X_{m0} \) represents the “do-nothing” alternative for management unit \( m \));

\( h_m \) = the first period in which management unit \( m \) is old enough to be harvested;

\( \lambda \) = the minimum area of mature forest habitat patch over all periods;

\( \mu \) = the maximum total perimeter of mature forest habitat patches over all periods;

\( M \) = the number of management units in the forest;

\( N \) = the number of pairs of management units in the forest that are adjacent;

\( T \) = the number of periods in the planning horizon;

\( c_{mt} \) = the discounted net revenue per hectare if management unit \( m \) is harvested in period \( t \), plus the discounted residual forest value based on the projected state of the stand at the end of the planning horizon;

\( A_m \) = the area of management unit \( m \) in hectares;

\( P_m \) = the perimeter of management unit \( m \) in meters;
\( CB_{pq} \) = the length of the common boundary between the two adjacent stands \( p, q \) in meters;

\( v_{mt} \) = the volume of sawtimber in m\(^3\)/hectare harvested from management unit \( m \) if it is harvested in period \( t \);

\( M_{ht} \) = the set of management units that are old enough to be harvested in period \( t \);

\( H_t \) = a continuous variable indicating the total volume of sawtimber in m\(^3\) harvested in period \( t \);

\( b_{lt} \) = a lower bound on decreases in the harvest level between periods \( t \) and \( t+1 \) (where, for example, \( b_{lt} = 1 \) requires non-declining harvests, or \( b_{lt} = 0.9 \) would allow a decrease of up to 10%);

\( b_{ht} \) = an upper bound on increases in the harvest level between periods \( t \) and \( t+1 \) (where, for example, \( b_{ht} = 1 \) allows no increase in the harvest level, or \( b_{ht} = 1.1 \) would allow an increase of up to 10%);

\( P \) = the set of all paths, or groups of contiguous management units, whose combined area is just above the maximum harvest opening size (the term “path,” as used in this paper, is defined in the following discussion);

\( P_i \) = the set of management units in path \( i \);
\( n_{p_i} \) = the number of management units in path \( i \);

\( h_i \) = the first period in which the youngest management unit in path \( i \) is old enough to be harvested;

\( O_{mt} \) = a binary variable whose value is 1 if management unit \( m \) meets the minimum age requirement for mature patches in period \( t \), i.e., the management unit is old enough to be part of a mature patch;

\( J_{mt} \) = the set of all prescriptions under which management unit \( m \) meets the minimum age requirement for mature patches in period \( t \);

\( C \) = the set of all clusters, or groups of contiguous management units whose combined area is just above the minimum large, mature patch size (the term “cluster,” as used in this paper, is defined in the following discussion);

\( S_c \) = the set of management units that compose cluster \( c \);

\( n_c \) = the number of management units in cluster \( c \);

\( B_{ct} \) = a binary variable whose value is 1 if all of the stands in cluster \( c \) meet the minimum age requirement for mature patches in period \( t \), i.e., the cluster is part of a mature patch;
\( BO_{mt} \) = a binary variable whose value is 1 if management unit \( m \) is part of a cluster that meets the minimum age requirement for large mature patches, i.e., the management unit is part of a patch that is big enough and old enough to constitute a large, mature patch;

\( \Omega^t_{pq} \) = a binary variable whose value is 1 if adjacent management units \( p \) and \( q \) are both part of a cluster that meets the minimum age requirement for large mature patches in period \( t \);

\( C_m \) = the set of all clusters that contain management unit \( m \);

\( Age^T_{mt} \) = the age of management unit \( m \) at the end of the planning horizon if it is harvested in period \( t \); and

\( \overline{Age}^T \) = the target average age of the forest at the end of the planning horizon.

Equation (1) specifies the first objective function of the problem, namely to maximize the discounted net revenue from the forest during the planning horizon, plus the discounted residual value of the forest. For age classes up to the optimal rotation, residual forest values are equal to the present value of the timber management costs and revenues on
the management unit, assuming that it will be harvested at the optimal economic rotation, plus the present value of the land expectation value (LEV) representing future rotations.

The LEV is the present value, per unit area, of the projected costs and revenues from an infinite series of identical even-aged forest rotations, starting initially from bare land. For age classes beyond the optimal economic rotation, residual forest values are equal to the liquidation value – i.e., the value of immediately harvesting the timber, plus the LEV for future rotations. Equation (2) maximizes the minimum amount of total area in large, mature forest patches over the time periods in the planning horizon. Equation (3) minimizes the maximum amount of total perimeter that borders the large, mature forest patches over the time periods in the planning horizon.

Constraint set (4) consists of logical constraints that allow only one prescription to be assigned to a management unit, including a do-nothing prescription. Harvest variables \( X_{mt} \) are only created for periods where the stand is old enough to be harvested.

Constraint set (5) consists of harvest accounting constraints that assign the harvest volume for each period to the harvest variables \( H_t \). Constraint sets (6) and (7) are flow constraints that restrict the amount by which the harvest level is allowed to change between
periods. In the example below, harvests were allowed to increase by up to 15% from one period to the next or to decrease by up to 3%.

Constraint set (8) consists of adjacency constraints generated with the Path Algorithm (McDill et al. 2002). These constraints limit the maximum size of a harvest opening, often necessary for legal or policy reasons, by prohibiting the concurrent harvest of any contiguous set of management units whose combined area just exceeds the maximum harvest opening size. The exclusion period imposed by these constraints equals one planning period, but the constraints can be modified easily to impose longer exclusion periods in integer multiples of the planning period. A “path” is defined for the purposes of the algorithm as a group of contiguous management units whose combined area just exceeds the maximum harvest opening size. These paths are enumerated with a recursive algorithm described in McDill et al. (2002). A constraint is written for each path to prevent the concurrent harvest of all of the management units in that path, since this would violate the maximum harvest opening size. This is done for each period in which it is actually possible to harvest all of the management units in a path. (In the initial periods of the planning horizon, some of the management units in a path may not be mature enough to be harvested.)
Constraint sets (9)–(15) are the mature patch size constraints. Constraint sets (9)-(10) determine whether or not management units meet the minimum age requirement for mature patches. These constraints sum over all of the prescription variables for a management unit under which the unit would meet the age requirement for mature patches in a given period. \( O_{mt} \) is equal to 1 if and only if one of these prescriptions has a value of 1, indicating that the management unit will be “old enough” in that period. One pair of these constraints is written for each management unit in each period.

Constraint sets (11)-(12) determine whether or not a cluster of management units meets the minimum age requirement for mature patches. Clusters are defined here as groups of contiguous management units whose combined area just exceeds the minimum mature patch size requirement. All possible clusters are enumerated using a recursive algorithm described in Rebain and McDill (2003b). A cluster meets the age requirement for mature patches in period \( t \) if all of the management units that compose that cluster meet the age requirement, as indicated by the \( O_{mt} \) variables for the management units in that cluster. \( B_{ct} \) takes a value of 1 if and only if cluster \( c \) meets the age requirement in period \( t \). These pairs of constraints are written for each cluster in each period.
Constraint sets (13)-(14) determine whether or not individual management units are part of a cluster that meets the minimum age requirement, i.e., whether a management unit is part of patch that is big enough and old enough. Since the clusters overlap, this constraint set is necessary to properly account for the total area of large, mature patch habitat. These constraints say that a management unit is part of a patch that meets the minimum age and size requirement for large, mature patches in period \( t \). \( BO_{wt} = 1 \) if and only if at least one of the clusters it belongs to meets the age requirement in that period.

Constraint set (15) specifies that the total mature patch area for each period must be larger than \( \lambda \) in all periods. Thus, \( \lambda \) cannot be larger than the area of large, mature forest patch habitat in any period. Equations (2) and (15) work together to capture the minimum amount of total area in the large, mature forest patches over all the time periods (the value of the variable \( \lambda \)) and maximize it.

Constraint sets (16)-(18) also work together. Constraint (16) calculates the total perimeter of all clusters that fulfill the minimum age and area requirement of mature forest patches, and assigns the maximum perimeter value to accounting variable \( \mu \). The maximum perimeter (\( \mu \)) is minimized by objective function (3). Constraints (17)-(18) define a new binary variable \( \Omega'_{pq} \) which substitutes a non-linear cross-product term.
These “linearization” constraints and variables avoid the cross-product term which would otherwise make the problem nonlinear. Notice that constraint set (18) is not necessary if objective function (3) is minimization. On the other hand, if maximizing the edge habitat is the objective, then constraints (18) are necessary and (17) could be dropped.

Constraint (19) is an ending age constraint. It requires the average age of the forest at the end of the planning horizon to be at least \( \bar{Age}^T \) years, preventing the model from over-harvesting the forest. In the example below, the minimum average ending age was set at 40 years, or \( \frac{1}{2} \) the optimal economic rotation.

Constraint sets (20)-(23) identify the stand prescription, mature patch size, and the cross-product linearization \( \Omega'_{pq} \) variables as binary.

The TOTALMIN Model:

The TOTALMIN Model is essentially the same as the MINIMAX Model, except the objective function (3) and constraint set (16) are replaced by (3’) and (16’), respectively:

\[
\text{Min } \sum_{t=T}^{T} \mu_t \quad \text{(3')} 
\]
\[
\sum_{m=1}^{M} P_{m}B_{m} - 2 \sum_{pq=1}^{N} CB_{pq} \Omega'_{pq} = \mu_{t} \quad \text{for } t = 1,2,\ldots,T
\] (16')

where \( \mu_{t} \) = the total perimeter of mature forest habitat patches in period \( t \).

Equation (3’) minimizes the sum of total perimeters that border the large, mature forest patches over the entire planning horizon.

Constraint (16’) calculates the total perimeter of all clusters that fulfill the minimum age and area requirements of mature forest patches in period \( t \), and assigns this value to accounting variable \( \mu_{t} \).

**Methods**

The impact of the MINIMAX and TOTALMIN models on the patch shape, temporal patch overlap and fragmentation of mature forest habitat was compared to that of a CONTROL model. The CONTROL model was formulated without perimeter minimization, i.e., a model like MINIMAX or TOTALMIN without objective function (3) or (3’). Constraints (16)–(18), however, were retained for bookkeeping purposes, i.e., to
tally the perimeter of the patches. The impacts were analyzed on a 50-stand hypothetical forest planning problem.

The hypothetical forest could be considered slightly over-mature, since approximately 40% of the area is between 60-100 years old and the optimal rotation is 80 years. The average stand size was 18 ha, and the total forest area was 900 ha. A 60-year planning horizon was considered, composed of three 20-yr periods. The four possible prescriptions for a given stand were: cut the management unit in period 1, period 2, or period 3, or do not cut it at all. Some stands had fewer prescriptions if they are too young to harvest in the early periods of the model. The minimum rotation age was 60 years. A maximum harvest opening size of 40 ha was imposed, and adjacent stands were allowed to be harvested concurrently as long as they did not violate this maximum opening size. All management units are smaller than the maximum harvest opening size. The wildlife species under consideration is assumed to need habitat patches that are at least 50 ha in size and at least 60 years old. Since the minimum habitat patch size is greater than the maximum harvest opening size, these patches must be composed of more than one management unit.
The impacts of the two perimeter minimizing formulations were examined at various levels of minimum habitat area. We wanted to know if there is any difference in improvement that can be made in terms of PAR and the number of patches for a mature forest habitat that is at least say 200 ha compared to one that is only 100 ha. These habitat levels, however, were not arbitrarily chosen. The set of Pareto-optimal solutions to the CONTROL model, which maximizes the NPV of the forest as well as the minimum amount of mature forest habitat over the planning periods in that forest, provided a convenient series of feasible minimum habitat levels. By definition, neither of the two objective function values (neither the NPV nor the minimum habitat area), corresponding to each Pareto-optimal solution for the CONTROL model, can be improved without compromising the other. The hypothetical forest planning problem, described above, yielded 36 such solutions, or harvest schedules. The attainment values on the objective that maximized the minimum amount of mature forest habitat served as minimum habitat levels for testing the perimeter minimizing formulations.

A multiple-objective optimization method, called “Alpha Delta” (Tóth et al. in press), was used to identify the set of Pareto-optimal solutions for the test problem. This algorithm was specifically developed to solve multiple-objective integer programs, such as
the models described in this paper. The two parameters of the algorithm, Alpha and Delta, were set to 1 degree and 0.01 ha, respectively. This setting ensured that a large number of Pareto-optimal solutions would be identified. For a detailed description of this algorithm and its parameters, please see Tóth et al. (in press).

After identifying the Pareto-optimal set, the following information was recorded for each solution: (1) NPV, (2) area of mature forest habitat in each planning period, (3) the total perimeter of the mature forest patches in each period, (4) the overlapping patch area between periods 1-2 and 2-3, and (5) the total number of patches that evolved over the planning horizon. This data was used to describe the shape, the temporal connectivity and the fragmentation of the mature forest habitat that evolved in the landscape using the CONTROL model.

The attribute of shape was measured by the PAR of the patches. One PAR value was calculated for each planning period. This was done by dividing the total perimeter of patches that form in that period by the total area of these patches. The average of these PARs was calculated for each of the Pareto-optimal solutions. The temporal overlaps of the patches between planning periods 1-2 and 2-3 were expressed as the percentage of the total patch area in period 1 and 2, respectively.
The fragmentation of mature forest habitat was accounted for by the number of patches that evolved during the entire planning horizon. The assumption behind describing fragmentation this way was that a mature forest habitat that consists of fewer patches with the same or greater total acreage is less fragmented.

The PARs and the number of patches corresponding to the CONTROL model solutions served as a baseline to improve upon using the MINIMAX and TOTALMIN models. As the latter two models are tri-objective integer programs (as opposed to the bi-objective CONTROL model), the number of Pareto-optimal solutions to the MINIMAX and TOTALMIN models might be different from that of the CONTROL model. We were interested only in solutions that were directly comparable to that of the CONTROL model in terms of the shape, temporal overlaps and the fragmentation of mature forest habitat. Thus, only a subset of the Pareto-optimal solutions to the MINIMAX and TOTALMIN models were identified. Starting from the solutions to the CONTROL model, the following 2-step procedure was used to find this subset of solutions.

Step 1 – the perimeter minimization phase – was to solve the following series of integer programs.
For $i = 1, \ldots, I$:

\[
\text{Min } \mu \quad \text{for the MINIMAX model} \quad (24), \text{ or }
\]

\[
\text{Min } \sum_{t \in T} \mu_t \quad \text{for the TOTALMIN model} \quad (24')
\]

subject to:

\[
\lambda \geq a_i \quad (25)
\]

where $I$ = the number of Pareto-optimal solutions to the CONTROL model;

$\mu$ = the maximum total perimeter of mature forest habitat patches over all periods;

$\mu_t$ = the total perimeter of mature forest habitat patches in period $t$

$\lambda$ = the minimum area of mature forest habitat patch over all periods;

$a_i$ = the attainment value on objective function (2) – the minimum area of mature forest patches – at Pareto-optimal solution $i$ of the CONTROL model.

The rest of the constraints were the same as (4)-(23) for both models, except (16), which was replaced by (16’) for the TOTALMIN model.
These IPs minimize the perimeter of the patches given a series of lower bounds \((a_i)\) on the minimum total area of the patches. Although the solutions to these programs can be used to assess the potential reduction in perimeter at each level of minimum habitat area, Step 2 is necessary to estimate how much NPV would have to be forgone to achieve these reductions.

In Step 2 – the NPV maximization phase – the following series of integer programs were solved.

For \(i = 1, \ldots, I: \)

\[
\text{Max } Z = \sum_{m=1}^{M} A_m [c_{m0} X_{m0} + \sum_{t=b_m}^{T} c_{mt} X_{mt}] 
\]

subject to:

\[
\lambda \geq a_i 
\]

\[
\mu \leq b_i 
\]

where \(Z\) is the discounted net revenue from the forest during the planning horizon, plus the discounted residual value of the forest;
\( b_i \) = the objective function value of solution \( i \) obtained by solving the integer programs described in Step 1.

Again, the rest of the constraints are the same as in Step 1.

The objective function values (NPV) of the solutions to the IPs in Step 2, using both the MINIMAX and the TOTALMIN formulations, were compared to those of the CONTROL model. The cost of perimeter minimization was expressed as the percentage difference between the pairs of NPVs for each minimum habitat level. In addition, the computational cost of the TOTALMIN formulation was contrasted to that of the MINIMAX formulation. This was done by comparing the solution times for both formulations at each minimum habitat level.

Minimizing the perimeter of the patches while requiring the minimum habitat area to be greater than or equal to a predefined level (i.e., using the MINIMAX or the TOTALMIN formulations) does not guarantee that the minimum PAR solution would be found. It only guarantees a minimum boundary solution. Solutions might exist that have longer boundaries but larger habitat areas, resulting in a lower PAR than that of the minimum boundary solution. Furthermore, the average cost (NPV forgone per unit PAR improvement) of the minimum boundary solution might be higher than some of the longer
boundary solutions. The decision maker (DM) might want to select a harvest scheduling regime that yields a certain amount of mature forest habitat in large and compact patches with minimal average costs. Exploring the trade-offs between the NPV and the perimeter at various levels of minimum habitat area would shed some light on the quality of the minimum boundary solution and might identify efficient (Pareto-optimal) alternatives with lower average costs.

In reflection to the above concerns, the trade-offs between the NPV and the perimeter of the mature forest patches were calculated by generating the set of Pareto-optimal solutions for these two objectives at three different levels of minimum habitat area. This was done by using the Alpha Delta algorithm to solve both the MINIMAX and the TOTALMIN models after changing objective function (2) to a constraint: $\lambda \geq a_j$, where the minimum habitat levels ($a_j$) were set to 64.3817 ($a_1$), 119.0577 ($a_2$), and 153.9933 ha ($a_3$), respectively. These values were evenly distributed along the set of minimum habitat levels generated by the CONTROL model. The parameters of the Alpha Delta algorithm, Alpha and Delta, were set to 1 degree and 1 meter, respectively. Again, this setting ensured that a large number of Pareto-optimal solutions would be found.
CPLEX 9.0 ([ILOG CPLEX 2002](#)) was used to generate the Pareto-optimal solutions to the above models and to solve the IPs in the perimeter minimization and NPV maximization phases. The program to automate the Alpha Delta algorithm was written in Microsoft Visual Basic 6 using the ILOG CPLEX Callable Libraries. The relative MIP gap tolerance parameter (optimality gap) was set to 0.00001 (0.001%), the MIP variable selection strategy parameter was set to ‘3’ (i.e., strong branching), and the strong branching parallel thread limit was set to 2. The reason for this setting was that, in the preliminary runs, parallel strong branching proved to be the most efficient variable selection strategy. All the IPs were solved on a Dual-AMD Athlon™ MP 2400+ (2.00 GHz) computer with 2.0 GB RAM.

**Results and discussion**

*Table 4-1* summarizes the impacts that the MINIMAX and TOTALMIN models had on the PAR, the number and the temporal overlap of mature forest patches. Clearly, the TOTALMIN models performed better in reducing the PAR and the number of patches, and in increasing the proportion of patch area that is part of the mature forest habitat both in periods 1-2 and 2-3. The downsides of this model are the higher costs and longer solution
times. As indicated by the maximum costs (roughly 10% for both models), however, we can also conclude that perimeter minimization can be expensive using either model.

Table 4-1. Comparison of the MINIMAX and TOTALMIN models

<table>
<thead>
<tr>
<th>Average (per problem)</th>
<th>NPV ($)</th>
<th>Cost (NPV forgone relative to that of the CONTROL model)</th>
<th>Temporal overlap between periods</th>
<th>P/A ratio (m/ha)</th>
<th>Number of patches</th>
<th>Solution time (minutes)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>mean</td>
<td>max</td>
<td>min</td>
<td>1-2</td>
<td>2-3</td>
</tr>
<tr>
<td>CONTROL</td>
<td>2,303,532</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>34.35%</td>
<td>37.21%</td>
</tr>
<tr>
<td>MINIMAX</td>
<td>2,229,338</td>
<td>3.47%</td>
<td>10.22%</td>
<td>1.55%</td>
<td>54.03%</td>
<td>62.29%</td>
</tr>
<tr>
<td>TOTALMIN</td>
<td>2,129,501</td>
<td>7.56%</td>
<td>10.57%</td>
<td>4.75%</td>
<td>59.04%</td>
<td>91.06%</td>
</tr>
</tbody>
</table>

The reason for lower PARs with the TOTALMIN approach is the following. As the MINIMAX model minimizes the maximum perimeter of the patches over the planning periods, it is not surprising that the improvements in PAR are restricted to those periods where the patch perimeter tends to be the largest. Once the perimeter of the patches in these "bottleneck" periods is minimized, there is no incentive for the model to improve the PARs in the other periods. It simply selects those harvest schedules that yield the highest NPVs while maintaining the predefined minimum acreage in mature forest habitat patches. In contrast, the TOTALMIN model tends to affect the shape of the patches in every period in an attempt to minimize the total perimeter over the entire planning horizon. The least
improvements are made in the first period, where the number of potential harvest schedules that could yield more compact patches is small. The only harvesting decision that could affect the shape of the patches in the first period is whether to cut a given unit or not in that period.

Figure 4-1 and the TOTALMIN ‘T’ model in Figure 4-2 illustrate how the two approaches, the TOTALMIN vs. MINIMAX models, work in an example problem. In this example, the required minimum amount of mature forest habitat was set to 154 ha. The first of the three numbers in each polygon on the maps is the harvest unit ID, the second is the initial age-class, and the third is the planning period when the unit is scheduled for harvest. As the planning periods are 20 years long in this example, a label of ‘6, 4, 3’ means that harvest unit 6, which is currently 60-80 years old (i.e., the 4th age class), is scheduled to be cut 40-60 years from now (in the 3rd period). If the last number is zero, the unit is not scheduled for harvest at any time during the planning horizon. The grey polygons in Figure 4-1 and 4-2 represent the harvest units that are parts of mature forest patches.
Figure 4-1. Mature forest habitat patches generated by the MINIMAX vs. CONTROL methods.
Figure 4-2. Mature forest patches generated by the TOTALMIN models

Although the MINIMAX model somewhat improved the PAR of the patches in each period (from 63.42 m/ha to 60.62 m/ha in period 1, from 68.77 to 59.27 in period 2 and from 61.84 to 59.44 in period 3), the change is barely visible on the maps (Figure 4-1). On
the other hand, the improvement in the compactness of the patches was striking with the TOTALMIN model in the second and third periods (Figure 4-2, left). The PAR was reduced to 40.86 m/ha in the second and third periods and no improvement was made in the first period. Similar degrees of improvement could be seen in the majority of the 35 example runs (Table 4-1). Also, it is worth observing in Figure 4-2 that the mature forest patch that has evolved in the second period using the TOTALMIN model has remained intact in the third period. Does minimizing the total perimeter result in static habitat patches? The data from 35 runs indicate that, on average, more than 90% of the area that is part of mature forest patches in period two is also part of the patches in period three (Table 4-1). This means that the patches did not change much from period two to three using the TOTALMIN model. This is good news for mature forest specialists that cannot move freely and rapidly to avoid habitat loss. To determine, however, whether these patches remain intact after the third planning period, models with longer planning horizons would need to be run. Unfortunately, the computational difficulty of IPs with four or more planning periods is still too daunting to conduct such experiments.
As the data in Table 4-1 indicate, the MINIMAX model is less likely to yield static patches. Only 62% of the area that is part of a mature forest patch in period two is also part of a mature patch in period three using this model.

The CONTROL model produced an average of 4.51 patches over the planning horizon. This number was significantly lower using both the MINIMAX (3.17) and the TOTALMIN (3.00) formulations (Table 4-1). This result is not surprising. Combining two or more disjoint patches into one, while keeping the total area of the patches above a predefined limit, generally yields a lower total perimeter. The above results imply that minimizing the perimeter not only tends to increase the temporal overlaps between the patches but is also likely to result in fewer and larger patches. Further experimentation is necessary to conclude, however, that the perimeter minimizing formulations proposed in this paper could serve as effective and computationally less expensive substitutes for models that directly enforce temporal or spatial connectivity (e.g., Önal and Briers in press or Briers 2002). The proposed models would have to be rigorously tested in a range of forest planning problems with various spatial structures. In certain instances, perimeter minimization can conflict with the spatial connectivity of the patches. For example,
preserving a forest patch with a high perimeter–area ratio might be desirable if that patch is needed to maintain connectivity.

We also hypothesized that the more mature forest habitat area is to be preserved in patches, the more improvement can be made in the PAR of these patches. Based on the data from the 35 experimental runs, indeed, there was a strong negative correlation (Pearson correlation = -0.965) between the average PAR of the patches and the minimum amount of mature forest habitat when the TOTALMIN model was used. No correlation (Pearson correlation = -0.29) occurred, however, between similar variables for the MINIMAX model. This finding also supports the conclusion that the TOTALMIN approach is more effective in pursuing compactness than the MINIMAX approach.

In sum, the TOTALMIN model appears to be more effective in reducing the PAR and the number of the patches and in increasing the temporal overlap. On the other hand, the MINIMAX approach is more cost efficient – both computationally and in allowing for more profitable harvest schedules. When the goal is to maintain a certain level of compactness in each planning period without the need to obtain optimally shaped patches, then the latter approach might be the way to go.
As it has been pointed out, minimizing the perimeter does not necessarily result in patches with minimum PARs. Minimizing the PAR directly, on the other hand, was not pursued here because the obvious formulation of this constraint would lead to non-linearity in the objective function. This non-linearity could be avoided by minimizing the perimeter- and maximizing the area of the patches simultaneously in a multiple-objective programming framework. A constraint that requires a small amount of minimum habitat area also needs to be added to avoid the zero perimeter - zero habitat scenario. From a harvest scheduling perspective, however, a third objective has to be incorporated in the model as well. This objective is to maximize the net revenues of the forest. The proposed approach, therefore, seeks the set of harvest schedules that are efficient with respect to the NPV and the perimeter of the patches at various levels of minimum habitat area. In this context, an efficient harvest schedule is one for which no other harvest schedule exists that would yield more net revenues or less perimeter without compromising the other.

The rest of the discussion focuses on analyzing the trade-offs between the NPV and the perimeter of the patches at three different levels of minimum habitat area: 64.3817, 119.0577, and 153.9933 ha. Figure 4-3 demonstrates these trade-offs for both the TOTALMIN and the MINIMAX models. Each point on the graphs represents an efficient
(or, equivalently, Pareto-optimal) harvest schedule. The lines that connect these points form the so-called efficient frontiers between the NPV and the perimeter of the patches at the three levels of minimum habitat area for the two approaches. These lines separate the regions where additional efficient harvest schedules are known not to exist from the regions where dominated (non-efficient) alternatives may exist (Tóth et al. in press). The rightmost points (in grey) on these efficient frontiers represent harvest schedules that were obtained by maximizing the NPV of the forest without regard to the perimeter of the patches while maintaining the predefined minimum habitat area requirements. The leftmost points (also in grey) are harvest schedules where the total perimeters (or maximum perimeters in the MINIMAX models) were minimized subject to the minimum habitat area requirements without regard to the NPV. Any point in between these two extremes is a compromise between the NPV and the perimeter.
Figure 4-3. Trade-offs between the NPV and the total perimeter of mature forest patches (TOTALMIN), and between the NPV and the maximum patch perimeter over the planning periods (MINIMAX)

These graphs can be useful for decision makers because they provide information about the nature of the trade-offs between the objectives in question. They can tell how much NPV would have to be forgone in order to reduce the perimeter of the patches to a
desired level. Similarly, they can tell how much more perimeter would have to be accepted to earn a given amount of net revenue. They can also aid the DM in eliminating solutions where a disproportionately high amount of one objective would have to be given up to gain minimal improvement on the other. For example, Point B on the bottom left graph could clearly serve as a preferred solution to Point T since the harvest schedule it represents yields almost the same total perimeter as Point T at a much lower (approx. $50,000 less) cost. The right hand side maps in Figure 4-2, labeled as “TOTALMIN ‘B’”, show the mature forest patches in grey that would evolve over the planning horizon if the harvest schedule corresponding to Point B was chosen (Figure 4-3). The patches on the left hand side maps in Figure 4-2, labeled as “TOTALMIN ‘T’”, would develop if Point T was chosen. The latter scenario would result in slightly more elongated patches in period two and three than the former one (Figure 4-2).

Another way of analyzing these alternative harvest schedules is to look at the average cost (i.e. average NPV forgone) of achieving one unit of PAR improvement over the solutions to the CONTROL model. The DM could easily eliminate the least cost-efficient alternatives from further consideration based on this statistic. Table 4-2 shows the
NPVs, PARs and average costs that correspond to each of the efficient harvest schedules that were generated by the Alpha-Delta method and using the TOTALMIN formulation.

Table 4-2. The cost of perimeter minimization with the TOTALMIN model

| Habitat >= 64.3817 ha | | Habitat >= 119.0577 ha | | Habitat >= 153.9933 ha |
|------------------------|------------------------|------------------------|------------------------|
| **NPV ($)** | **Average P/A Ratio (m/ha)** | **Average Cost per unit PAR imp.** | **NPV ($)** | **Average P/A Ratio (m/ha)** | **Average Cost per unit PAR imp.** | **NPV ($)** | **Average P/A Ratio (m/ha)** | **Average Cost per unit PAR imp.** |
| 1 | 2,427,424 | 70.71 | N/A | 2,372,633 | 67.5 | N/A | 2,238,791 | 64.68 | N/A |
| 2 | 2,406,217 | 70.16 | $38,028 | 2,354,665 | 67.09 | $44,925 | 2,236,222 | 62.61 | $1,242 |
| 3 | 2,376,803 | 64.50 | $8,142 | 2,321,840 | 55.30 | $14,145 | 2,219,769 | 59.56 | $1,242 |
| 4 | 2,373,246 | 66.88 | $14,145 | 2,315,861 | 61.14 | $8,937 | 2,215,169 | 58.44 | $1,242 |
| 5 | 2,369,735 | 61.75 | $6,434 | 2,312,124 | 61.98 | $10,960 | 2,213,536 | 57.27 | $1,242 |
| 6 | 2,359,213 | 61.26 | $7,211 | 2,308,129 | 55.47 | $5,362 | 2,212,977 | 56.51 | $1,242 |
| 7 | 2,333,438 | 58.99 | $8,019 | 2,299,392 | 59.87 | $9,806 | 2,209,136 | 55.92 | $1,242 |
| 8 | 2,329,005 | 60.50 | $9,632 | 2,286,700 | 60.58 | $12,144 | 2,181,014 | 54.48 | $1,242 |
| 9 | 2,312,173 | 60.05 | $10,805 | 2,276,818 | 57.99 | $10,080 | 2,175,380 | 52.60 | $1,242 |
| 10 | 2,269,390 | 56.66 | $9,528 | 2,263,789 | 56.66 | $10,047 | 2,168,379 | 53.19 | $1,242 |
| 11 | 2,243,177 | 53.96 | $9,564 | 2,243,177 | 53.96 | $9,564 | 2,165,597 | 50.41 | $1,242 |
| 12 | 2,212,710 | 54.53 | $12,336 | 2,212,710 | 54.53 | $12,336 | 2,139,321 | 51.50 | $1,242 |
| 13 | 2,204,132 | 51.21 | $10,344 | 2,204,132 | 51.21 | $10,344 | 2,139,242 | 50.46 | $1,242 |
| 14 | 2,136,270 | 50.39 | $7,176 | 2,136,270 | 50.39 | $7,176 | 2,115,789 | 49.24 | $1,242 |
| 15 | 2,115,789 | 49.24 | $7,969 | 2,113,340 | 49.18 | $8,095 | 2,102,264 | 48.15 | $1,242 |
| 16 | 2,058,601 | 48.38 | $11,056 | 2,058,601 | 48.38 | $11,056 | 2,046,132 | 47.24 | $1,242 |

The table displays three separate series of data representing the three levels of minimum habitat requirements. The first entries at the top (in bold) are the harvest schedules that were obtained without perimeter minimization (by the CONTROL model). They correspond to the leftmost points on the left hand side charts in Figure 4-3. The last entries (also in bold) were obtained by minimizing the perimeter given the three minimum habitat levels. They correspond to the rightmost points on the left hand side charts in
There is a one on one correspondence between the entries of the table and the points on the charts representing the compromise harvest schedules between the NPV and the perimeter of the patches.

One observation that merits attention is that harvest schedule #20 in the rightmost table yields patches with a lower average PAR than that of #21, the minimum boundary solution. Furthermore, its average cost per PAR improvement is also lower. This confirms the hypothesis that perimeter minimization does not necessarily generate patches with minimum PARs. Second, the strong fluctuation of the average costs along the efficient frontiers also imply that good alternative solutions might be overlooked if only the minimum boundary solution is identified.

Clearly, finding the best compromise forest management regime for a given landscape is not trivial if several conflicting objectives are present. Balancing the inherent trade-offs between these objectives requires the forest planners and decision makers to examine the problem from different angles.
Improving the integer programming formulations

The integer programs discussed in this paper are hard to solve. In fact, they are \( \mathcal{NP} \)-Hard, as they build on the area restriction model (ARM, Murray 1999), a generalization of the unit restriction model (URM, Murray 1999). Since the URM is an equivalent of the maximum weight stable set problem (SSP) which is known to be \( \mathcal{NP} \)-Hard, the MINIMAX and TOTALMIN models are also \( \mathcal{NP} \)-Hard. Even problems of moderate size (100 stands or larger) can be prohibitively difficult to solve to exact optimality. This is why it might be worth looking into potential ways of improving the formulation of these models.

Rebain and McDill (2003b) proposed to substitute constraint sets (9) and (11) with:

\[
\sum_{m \in \mathcal{M}_p} \sum_{j \in \mathcal{J}_{mc}} x_{mj} - n_{c}B_{ct} \geq 0 \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots, T \tag{29}
\]

This way, binary variables \( O_{mt} \) could be eliminated from the model. Constraint set (29), in turn, could be strengthened by using:

\[
\sum_{j \in \mathcal{J}_{mt}} x_{mj} \geq B_{ct} \quad \text{for } m \in \mathcal{S}_c, \text{ for } c \in C, \text{ and } t = 1, 2, \ldots, T \tag{29'}
\]
Although this move would require more constraints, (29') is stronger than (29), because, without proof, 
\[
\left\{ \sum_{j \in J_{ct}} x_{mj} \geq B_{ct} \right\} \subseteq \left\{ \sum_{m \in S_c, j \in J_{ct}} x_{mj} - n C B_{ct} \geq 0 \right\} \quad \text{where } X_{mj}, B_{ct} \in R.
\]
To ensure, however, that \( B_{ct} = 1 \) if and only if \( \sum_{j \in J_{ct}} x_{mj} = 1 \) for \( \forall m \in S_c \), then constraint sets (10) and (12) would have to be replaced by:

\[
\sum_{m \in S_c, j \in J_{ct}} x_{mj} - B_{ct} \leq 1 \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots, T \tag{30}
\]

In summary, constraints (29’), (30) can replace constraints (9)-(12) forming a tighter formulation.

Finally, using similar strengthening techniques, constraint sets (14) and (17) can be substituted by

\[
BO_{mt} \geq B_{ct} \quad \text{for } c \in C, m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \tag{31}
\]

and by

\[
BO_{pi} \geq \Omega^i_{pq}, BO_{qi} \geq \Omega^i_{pq} \quad \text{for } t = 1, 2, \ldots T, \text{ and } pq = 1, 2, \ldots N \tag{32}
\]

respectively. \textit{Fischer and Church (2003)} used constraints of form (32) to identify minimum boundary reserve networks.
As a final note, constraints (10), (12) or (30), and (14) can be dropped from the model if the mature forest habitat area is maximized in the objective function (e.g. Rebain and McDill 2003a, b). In these cases, the sense of the objective function ensures that the \( BO_{mf} \) variables take the value of one whenever it is feasible. In the perimeter minimizing formulations, however, the above constraints could not be left out; otherwise, the models would have no incentive to “turn on” the \( BO_{mf} \) variables even if the conditions are met to do so. This would result in incorrectly accounting for the perimeter of the patches.

Although, in general, tighter IP formulations solve faster, thorough testing is needed to confirm whether the above changes would, indeed, yield shorter solution times. Also, as some commercial IP solvers limit the number of constraints that can be used in a given problem, some forest modelers might have to use the weaker but less “constraint-intensive” formulation.

**Conclusions**

The models presented in this article enable forest planners to control the size and shape of the mature forest patches that can evolve on a landscape as a result of various
harvest schedules. At the same time, they assure that the total area of contiguous harvests never exceed a predefined limit. This means that even though maximum harvest opening constraints tend to disperse harvests and compromise the development of large compact patches, this negative impact can be kept at bay effectively through spatially-explicit harvest scheduling. This accomplishment, however, comes with a significant computational cost. The integer programming model is more complex and harder to solve to optimality, particularly if applied to larger problems. There are several reasons, however, to be optimistic about the future use of these models. First, as it had been pointed out, there is a lot of room for improving the formulation itself. Second, computers and software packages are constantly getting faster and more powerful. Third, although enabling the harvest scheduling model to develop mature forest patches requires a lot of new constraints and variables, the marginal computational cost of incorporating restrictions on the spatial attributes of these patches is small. This research has demonstrated that minimizing the perimeter of the patches not only makes their shape more compact, but also increases their temporal connectivity and yields fewer and larger patches at the same time.

Generating efficient harvest schedules with respect to the NPV of the forest, the minimum amount of mature forest habitat in large patches and the perimeter of these
patches is one potential use of the proposed perimeter minimizing formulations. This approach can help the DM select his or her best compromise solution after having seen several alternatives. Setting up restrictions or goals on the size, shape and other spatial attributes of patches might not be viable prior to the planning process because of the difficulty in quantifying these parameters upfront. The multiple-objective approach proposed in this paper enables the DM to see what is possible before determining what should be. Finding the set of efficient harvest schedules with respect to the three objectives, however, is not trivial. The approach followed by this study can only identify a small subset. This is true even if the NPV-perimeter tradeoffs are generated at more than three levels of minimum habitat area. Although, generating the complete set of tradeoffs between the three objectives could provide the DM with more information, this benefit might be offset by the increased computational expense.

Analyzing the impacts of the perimeter minimizing formulations on the spatial connectivity of mature forest patches within landscapes of various spatial structures is a subject of future research. Further research is also needed to identify the “best” integer programming formulation, the one that yields the shortest solution times, of the patch size and patch perimeter attributes.
Chapter 5. Finding the Efficient Frontier of a Tri-Criteria, Spatially-explicit, Harvest Scheduling Problem

Sándor F. Tóth and Marc E. McDill

Abstract: Public forests have many conflicting uses. Designing forest management schemes that provide the public with an optimal bundle of benefits is therefore a major challenge to forest planners. Spatially-explicit harvest scheduling models, designed to optimize timber production, can be used to promote non-timber uses as well. Although quantifying and visualizing the trade-offs between the various uses can be useful to decision makers, doing so is not trivial if three or more conflicting objectives are present and the solution alternatives are discrete. This study extends four methods of evaluating the trade-offs between two objectives by generating sets of Pareto-efficient solutions to enable them to handle three or more objectives. The Alpha-Delta, the ε-Constraining, the Weighted Objectives and the Tchebycheff methods were tested on a 50-stand hypothetical forest planning problem with three objectives: 1) maximizing the net revenues of the forest, 2) maximizing the area of evolving mature forest patches, and 3) minimizing the perimeter of these patches. The comparison criteria were the number of Pareto-optimal solutions found in a given time limit, the user’s level of control to filter the solutions, and the ease of
handling the general, $n$-objective case. Although the Alpha-Delta method appeared to be the simplest to generalize and to filter the solutions in line with the decision maker’s interests, the Weighted Objectives Method found more Pareto-optimal solutions. Some important numerical issues related to the use of integrality and optimality tolerance gaps in multiple-criteria optimization are also discussed.

**Key words**: Multiple-criteria optimization, trade-offs between timber management and wildlife habitat, 0-1 programming

**Introduction**

Management planning problems with conflicting objectives occur frequently in forestry. The public expects more from forest resources than merely timber production, including watershed protection, wildlife habitat management, recreation, carbon sequestration and grazing. Stakeholder groups such as the timber industry or environmental organizations often hold strongly conflicting values related to these uses, and conflicts between timber and non-timber objectives are common. Harvesting can fragment sensitive habitats, obstruct the movement of wildlife, increase fire risk, and
reduce the aesthetic value of the forest. Recreation can also put enormous stress on forest ecosystems, and eliminating timber production from public forests, as many suggest, only increases the pressure on other forest resources, as timber consumption is increasing (Thomas 2000). In some cases, the many objectives of forest management are so incompatible that only zoning – setting aside separate forest blocks for the various uses – can solve the problem (Binkley 2000). In other cases, the trade-offs between the conflicting goals can be balanced effectively within the same landscape or forest through careful planning (Rosenbaum 2000). In most cases, quantifying the trade-offs to find out exactly how incompatible certain forest uses are can help decision makers (DM) select the best compromise management alternative.

The spatial layout of forestry operations such as harvesting or road construction can have a profound impact on many non-timber objectives. Spatial optimization techniques such as harvest scheduling models have proven to be useful in designing the location and timing of these operations while also addressing wildlife habitat concerns (e.g., Rebain and McDill 2003a,b). Spatially-explicit harvest scheduling models, usually built as integer programs, are used to determine when each harvest unit should be cut in order to balance the various forest uses. Most often, these models have been formulated as single objective
problems, where one forest use is optimized subject to a range of restrictions (e.g.,

or Rebain and McDill 2003a, b). Some of these restrictions ensure that minimum
requirements on the other uses are met. One example would be to maximize timber output
or the net revenues from a forest subject to constraints requiring a balanced ending age-
class distribution, a smooth flow of timber production over time, and maintaining a
minimum amount of mature forest habitat in large compact patches while never exceeding
the maximum harvest opening size. Alternatively, the amount of mature forest habitat
could be maximized subject to minimum net present value or minimum timber output
constraints. In these types of formulations, the DM(s) have to specify the minimum
requirements on some forest uses prior to the planning process. Defining harvest target
constraints might be relatively straightforward, but setting limits on the perimeter of mature
forest habitat patches, for instance, might not. Without knowing what perimeter restrictions
are feasible and which are too modest, specifying the limits is typically guesswork and can
lead to poor decisions. Another frequently used approach, goal programming, does not
completely overcome this problem either. This approach requires the DM to set up targets
on the objectives that, unlike constraints, do not have to be met. Goal programs (GP)
minimize the deviations from the targets in either an order of preference (preemptive GP) or in line with a set of weights assigned to each objective (non-preemptive GP). In either way, the DM has to specify both the targets and the weights or a preference list for the objectives prior to the planning process.

When possible, generating and visualizing the complete set or a filtered set of efficient solutions might help the DM(s) acquire a holistic view of the problem to enable a more informed decision when selecting the best compromise management alternatives (Tóth et al. in press). A solution (i.e., a forest management alternative) is efficient or Pareto-optimal (Pareto 1909) if none of the objective function values corresponding to that solution can be improved without compromising the others. Solutions that are not Pareto-optimal are “dominated” and are clearly inferior as long as all objectives are represented in the model. One solution dominates another if it has a strictly better attainment of at least one objective while the attainments of the rest of the objectives are not worse than that of the other solution (Koopmans 1951). The set of efficient solutions define the efficient frontier in a coordinate system where the axes represent the objective function values. This coordinate system is called the “objective space.” Studying this frontier can be valuable to the DM(s) for two reasons. First, the efficient frontier separates the region where additional
solutions do not exist from the region where dominated solutions might exist (Tóth et al. in press). Thus, the DM can assess the limits of simultaneously achieving several conflicting objectives. In other words, the efficient frontier can answer the question: what is possible? Second, by moving along this frontier (i.e., by moving between Pareto optimal solutions), one can also assess the amount of one goal that must be forgone in order to achieve a given increase in the amount of another goal (Tóth et al. in press). Strictly speaking, the efficient frontier demonstrates the set of trade-offs between competing objectives.

Since in spatially-explicit harvest scheduling model the primary type of management decision is whether to cut a given unit within a given time interval or not, the set of feasible management alternatives is discrete. As a result, the set of attainable objective function values and, hence, the efficient frontier itself are neither convex nor continuous. This important geometric property makes it an immense computational challenge to generate the efficient frontiers for spatially-explicit harvest scheduling problems.

Tóth et al. (in press) evaluated and tested four traditional methods of generating the efficient frontier for a bi-objective spatially-explicit harvest scheduling problem. The four approaches, (1) the Weighted Objective Function method (Geoffrion 1968), (2) the \( \varepsilon \)-
Constraining method (Sadagopan et al. 1982), (3) the Decomposition method based on the Tchebycheff-Metric (Eswaran et al. 1989), and (4) the Triangles method (Chalmet et al. 1986) were tested on a 50-stand hypothetical forest planning problem. Tóth et al. (in press) also proposed a new approach called the “Alpha-Delta” method that compared well against the other approaches. The two objectives of interest in their test case were to maximize the net present value (NPV) of the forest and to maximize the minimum amount of mature forest habitat in large patches over the planning horizon.

A limited number of theoretical studies on generating the set of efficient alternatives for three- or more objective integer programs have been done. The primary area of research has been the family of the so-called reference point methods (Ehrgott and Wiecek 2005). The concept is simple: an efficient solution can be found by minimizing the distance between a reference point, which can be any non-attainable solution in the objective space (such as the ideal solution or targets for a goal program), and potential non-dominated solutions. The distance measure that is used most often is the weighted (or not) Tchebycheff-Metric. Different efficient solutions can be found by either varying the weights on the metric while keeping the reference point constant (e.g., Eswaran et al. 1989), or by varying the reference points while keeping the weights constant (e.g., Alves and ...
Due to the discrete nature of integer programming, different weight combinations and different reference points can lead to identical solutions. Thus, a “smart” decomposition of the weight space (or the reference point space) into regions that lead to the same solutions is needed to reduce the time spent finding redundant solutions. These regions are called *indifferent sets* and are convex in the weight space and non-convex in the reference space (*Alves and Clímaco 2001*). The reference point methods find efficient solutions by determining or approximating these indifference sets.

Case studies assessing the numerical and computational performance of the available methods as applied to larger than “illustrative” problem instances are not common. This is particularly true for the area of forest resources and wildlife management. Although efficient frontiers with respect to two objectives have been studied in *Roise et al. (1990)*, *Holland et al. (1994)*, *Cox and Sullivan (1995)*, *Arthaud and Rose (1996)*, *Church et al. (1996)* and (2000), *Snyder and Revelle (1997a)*, *Williams (1998)*, and *Richards and Gunn (2000)*, the efficient frontiers of discrete forest management decision problems with three or more objectives have not apparently been researched.

The present study builds on *Tóth et al. (in press)* by extending three of the four traditional approaches, the Weighted Objective Function method, the $\varepsilon$-Constraining
method, the Decomposition method based on the Tchebycheff-Metric, and the proposed Alpha-Delta method to handle three or more objectives. The same 50-stand hypothetical forest planning problem that was used in Tóth et al. (in press) is used here to demonstrate the mechanics, and to test the performance of the extended methods in generating the efficient frontier of a tri-objective spatially-explicit harvest scheduling problem. Two of the objectives are the same as in Tóth et al. (in press). The third is to minimize the total perimeter of the evolving mature forest patches over the planning horizon. The formulation of this objective was introduced in Chapter 4. Maximizing the area of mature forest habitat patches while minimizing their total perimeter promotes forested landscapes with several desirable spatial characteristics. This approach fosters the development of patches with low perimeter-area ratios, increases their temporal overlap, and results in fewer and larger patches.

Clearly, finding ways to generate the trade-offs between three or more objectives can be important in spatial, landscape-level forest planning because of the numerous objectives involved. Furthermore, while in the bi-criteria case, the efficient frontier can be represented in two dimensions, visualizing the trade-offs between three or more objectives requires a different approach. Multiple options should be explored while bearing
in mind the differences between DMs, as one approach might work for one individual but not for another.

Visualization of trade-offs is not the only issue that emerges as one makes the leap from two- to three-objective optimization. Several computational and numerical problems also become evident that can be critical in obtaining accurate solutions. Failure to treat these problems appropriately may lead to poor decision-making. This work addresses these issues in detail.

**The Model Formulation**

This section describes a tri-criteria integer programming formulation that (1) maximizes the net present value of the forest, (2) maximizes the minimum amount of mature forest habitat in large patches over the planning horizon, and (3) minimizes the total length of the edges of these patches over the planning horizon. This perimeter minimization approach was introduced as the “TOTALMIN Model” in Chapter 4. The model includes harvest flow constraints, maximum harvest opening size constraints, constraints that define the minimum area of large, mature patch habitat over time, and a minimum average ending age constraint. The model formulation of the mature forest patch
criterion is a slightly modified version of the one presented in Rebain and McDill (2003a).

Formulation of the maximum harvest area constraints is a generalization of the formulation presented in McDill et al. (2002).

\[
Max \quad Z = \sum_{m=1}^{M} A_m [c_m X_{m0} + \sum_{t=h_m}^{T} c_{mt} X_{mt}] 
\]

\[\text{Max} \quad \lambda \] \hspace{2cm} (1)

\[\text{Min} \quad \sum_{t \in T} \mu_t \] \hspace{2cm} (2)

subject to:

\[X_{m0} + \sum_{t=h_m}^{T} X_{mt} \leq 1 \quad \text{for } m = 1, 2, \ldots M \] \hspace{2cm} (4)

\[\sum_{m \in M_t} v_{mt} \cdot A_m \cdot X_{mt} - H_t = 0 \quad \text{for } t = 1, 2, \ldots T \] \hspace{2cm} (5)

\[b_{ht} H_t - H_{t+1} \leq 0 \quad \text{for } t = 1, 2, \ldots T-1 \] \hspace{2cm} (6)

\[-b_{ht} H_t + H_{t+1} \leq 0 \quad \text{for } t = 1, 2, \ldots T-1 \] \hspace{2cm} (7)

\[\sum_{m \in P_t} X_{mt} \leq n_P - 1 \quad \text{for all } P_t \in P \text{ and } t = h_t, \ldots, T \] \hspace{2cm} (8)
\[ \sum_{j=J_{mt}} X_{mj} - O_{mt} \geq 0 \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \] (9)

\[ \sum_{j=J_{mt}} X_{mj} - |J_{mt}|O_{mt} \leq 0 \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \] (10)

\[ \sum_{m \in S_j} O_{mt} - n_j B_{ct} \geq 0 \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots, T \] (11)

\[ \sum_{m \in S_j} O_{mt} - B_{ct} \leq n_c - 1 \quad \text{for } c \in C, \text{ and } t = 1, 2, \ldots, T \] (12)

\[ \sum_{c \in C_w} B_{ct} - BO_{mt} \geq 0 \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \] (13)

\[ \sum_{c \in C_w} B_{ct} - |C_m|BO_{mt} \leq 0 \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 1, 2, \ldots, T \] (14)

\[ \sum_{m=1}^{M} A_{mm} BO_{mt} \geq \lambda \quad \text{for } t = 1, 2, \ldots T \] (15)

\[ \sum_{m=1}^{M} P_{mm} BO_{mt} - 2 \sum_{pq=1}^{N} CB_{pq} \Omega_{pq}^{t} = \mu_{t} \quad \text{for } t = 1, 2, \ldots T \] (16)

\[ BO_{pt} + BO_{qt} - 2\Omega_{pq}^{t} \geq 0 \quad \text{for } t = 1, 2, \ldots T, pq = 1, 2, \ldots, N \] (17)

\[ BO_{pt} + BO_{qt} - \Omega_{pq}^{t} \leq 1 \quad \text{for } t = 1, 2, \ldots T, pq = 1, 2, \ldots, N \] (18)

\[ \sum_{m=1}^{M} \left[ (Age_{mt}^{T} - \overline{Age}_{mt}^{T})X_{mt} + \sum_{t=h}^{T} (Age_{mt}^{T} - \overline{Age}_{mt}^{T})X_{mt} \right] \geq 0 \] (19)

\[ X_{mt} \in \{0, 1\} \quad \text{for } m = 1, 2, \ldots M, \text{ and } t = 0, h_{m}, h_{m} + 1, \ldots, T \] (20)

\[ B_{ct} \in \{0, 1\} \quad \text{for } c \in C, \ t = 1, 2, \ldots, T \] (21)
\[ O_{mt}, BO_{mt}, BO_{pt}, BO_{qt} \in \{0,1\} \quad \text{for } m, p, q = 1, 2, \ldots M, t = 1, 2, \ldots T \quad (22) \]

\[ \Omega'_{pq} \in \{0,1\} \quad \text{for } pq \in N \quad (23) \]

where \( X_{mt} \) = a binary decision variable whose value is 1 if management unit \( m \) is to be harvested in period \( t \) for \( t = h_m, h_{m+1}, \ldots, T \); when \( t = 0 \), the value of the binary variable is 1 if management unit \( m \) is not harvested at all during the planning horizon (i.e., \( X_{m0} \) represents the “do-nothing” alternative for management unit \( m \));

\( h_m \) = the first period in which management unit \( m \) is old enough to be harvested;

\( \lambda \) = the minimum area of mature forest habitat patch over all periods;

\( \mu_t \) = the total perimeter of mature forest habitat patches in period \( t \);

\( M \) = the number of management units in the forest;

\( N \) = the number of pairs of management units in the forest that are adjacent;

\( T \) = the number of periods in the planning horizon;

168
\( c_{mt} \) = the discounted net revenue per hectare if management unit \( m \) is harvested in period \( t \), plus the discounted residual forest value based on the projected state of the stand at the end of the planning horizon;

\( A_m \) = the area of management unit \( m \) in hectares;

\( P_m \) = the perimeter of management unit \( m \) in meters;

\( CB_{pq} \) = the length of the common boundary between the two adjacent stands \( p, q \) in meters;

\( v_{mt} \) = the volume of sawtimber in \( m^3/\)hectare harvested from management unit \( m \) if it is harvested in period \( t \);

\( M_{ht} \) = the set of management units that are old enough to be harvested in period \( t \);

\( H_t \) = a continuous variable indicating the total volume of sawtimber in \( m^3 \) harvested in period \( t \);

\( b_{lt} \) = a lower bound on decreases in the harvest level between periods \( t \) and \( t+1 \) (where, for example, \( b_{lt} = 1 \) requires non-declining harvests, or \( b_{lt} = 0.9 \) would allow a decrease of up to 10%).
$b_{ht}$ = an upper bound on increases in the harvest level between periods $t$ and $t+1$

(where, for example, $b_{ht} = 1$ allows no increase in the harvest level, or $b_{ht} = 1.1$ would allow an increase of up to 10%);

$P$ = the set of all paths, or groups of contiguous management units, whose combined area is just above the maximum harvest opening size (the term “path,” as used in this paper, is defined in the following discussion);

$P_i$ = the set of management units in path $i$;

$n_{P_i}$ = the number of management units in path $i$;

$h_i$ = the first period in which the youngest management unit in path $i$ is old enough to be harvested;

$O_{mt}$ = a binary variable whose value is 1 if management unit $m$ meets the minimum age requirement for mature patches in period $t$, i.e., the management unit is old enough to be part of a mature patch;

$J_{mt}$ = the set of all prescriptions under which management unit $m$ meets the minimum age requirement for mature patches in period $t$;
\( C \) = the set of all clusters, or groups of contiguous management units whose combined area is just above the minimum large, mature patch size (the term “cluster,” as used in this paper, is defined in the following discussion);

\( S_c \) = the set of management units that compose cluster \( c \);

\( n_c \) = the number of management units in cluster \( c \);

\( B_{ct} \) = a binary variable whose value is 1 if all of the stands in cluster \( c \) meet the minimum age requirement for mature patches in period \( t \), i.e., the cluster is part of a mature patch;

\( BO_{mt} \) = a binary variable whose value is 1 if management unit \( m \) is part of a cluster that meets the minimum age requirement for large mature patches, i.e., the management unit is part of a patch that is big enough and old enough to constitute a large, mature patch;

\( \Omega^t_{pq} \) = a binary variable whose value is 1 if adjacent management units \( p \) and \( q \) are both part of a cluster that meets the minimum age requirement for large mature patches in period \( t \);

\( C_m \) = the set of all clusters that contain management unit \( m \);
Age^T_{m} = \text{the age of management unit } m \text{ at the end of the planning horizon if it is harvested in period } t; \text{ and}

\bar{Age}^T = \text{the target average age of the forest at the end of the planning horizon.}

Equation (1) specifies one of the three objective functions of the problem, namely to maximize the discounted net revenue from the forest during the planning horizon, plus the discounted residual value of the forest. For age classes up to the optimal rotation, residual forest values are equal to the present value of the timber management costs and revenues on the management unit, assuming that it will be harvested at the optimal economic rotation, plus the present value of the land expectation value (LEV) representing future rotations. The LEV is the present value, per unit area, of the projected costs and revenues from an infinite series of identical even-aged forest rotations, starting initially from bare land. For age classes beyond the optimal economic rotation, residual forest values are equal to the liquidation value – i.e., the value of immediately harvesting the timber, plus the LEV for future rotations. Equation (2) maximizes the minimum amount of total area in large, mature forest patches over the time periods in the planning horizon. Equation (3)
minimizes the sum of the perimeters of the large, mature forest patches over the entire planning horizon.

Constraint set (4) consists of logical constraints that allow only one prescription to be assigned to a management unit, including a do-nothing prescription. Harvest variables \( X_{mt} \) are only created for periods where the stand is old enough to be harvested. Constraint set (5) consists of harvest accounting constraints that assign the harvest volume for each period to the harvest variables \( H_t \). Constraint sets (6) and (7) are flow constraints that restrict the amount by which the harvest level is allowed to change between periods. In the example below, harvests were allowed to increase by up to 15% from one period to the next or to decrease by up to 3%.

Constraint set (8) consists of adjacency constraints generated with the Path Algorithm (McDill et al. 2002). These constraints limit the maximum size of a harvest opening, often necessary for legal or policy reasons, by prohibiting the concurrent harvest of any contiguous set of management units whose combined area just exceeds the maximum harvest opening size. The exclusion period imposed by these constraints equals one planning period, but the constraints can be modified easily to impose longer exclusion periods in integer multiples of the planning period. A “path,” more properly called a
“cover” (see Chapter 2) is defined for the purposes of the algorithm as a group of contiguous management units whose combined area just exceeds the maximum harvest opening size. These covers are enumerated with a recursive algorithm described in McDill et al. (2002). A constraint is written for each cover to prevent the concurrent harvest of all of the management units in that cover, since this would violate the maximum harvest opening size. This is done for each period in which it is actually possible to harvest all of the management units in a cover. (In the initial periods of the planning horizon, some of the management units in a cover may not be mature enough to be harvested.)

Constraint sets (9)–(15) are the mature patch size constraints. Constraint sets (9)-(10) determine whether or not management units meet the minimum age requirement for mature patches. These constraints sum over all of the prescription variables for a management unit under which the unit would meet the age requirement for mature patches in a given period. $O_{mr}$ is equal to 1 if and only if one of these prescriptions has a value of 1, indicating that the management unit will be “old enough” in that period. One pair of these constraints is written for each management unit in each period.

Constraint sets (11)-(12) determine whether or not a cluster of management units meets the minimum age requirement for mature patches. Clusters are defined here as
groups of contiguous management units whose combined area just exceeds the minimum mature patch size requirement. All possible clusters are enumerated using a recursive algorithm described in Rebain and McDill (2003b). A cluster meets the age requirement for mature patches in period \( t \) if all of the management units that compose that cluster meet the age requirement, as indicated by the \( O_{mt} \) variables for the management units in that cluster. \( B_{ct} \) takes a value of 1 if and only if cluster \( c \) meets the age requirement in period \( t \).

These pairs of constraints are written for each cluster in each period.

Constraint sets (13)-(14) determine whether or not individual management units are part of a cluster that meets the minimum age requirement, i.e., whether a management unit is part of patch that is big enough and old enough. Since the clusters overlap, this constraint set is necessary to properly account for the total area of large, mature patch habitat. These constraints say that a management unit is part of a patch that meets the minimum age and size requirement for large, mature patches in period \( t \) (\( BO_{mt} = 1 \)) if and only if at least one of the clusters it belongs to meets the age requirement in that period.

Constraint set (15) specifies that the total mature patch area for each period must be larger than \( \lambda \) in all periods. Thus, \( \lambda \) cannot be larger than the area of large, mature forest patch habitat in any period. Equations (2) and (15) work together to capture the minimum
amount of total area in the large, mature forest patches over all the time periods (the value of the variable $\lambda$) and maximize it.

Constraint sets (16)-(18) also work together. Constraint (16) calculates the total perimeter of all clusters that fulfill the minimum age and area requirements of mature forest patches in period $t$, and assigns this value to accounting variable $\mu_t$. The sum of the total perimeters over the planning horizon is minimized by objective function (3). Constraints (17)-(18) define a new binary variable $\Omega'_{pq}$ that substitutes for what would otherwise be a non-linear cross-product term ($\Omega'_{pq} = BO_{pq} BO_{qu}$) in (16). Constraint set (18) is not necessary if objective function (3) is to minimize the perimeter. On the other hand, if maximizing edge habitat were the objective, then constraint set (18) would be necessary and (17) could be dropped.

Constraint (19) is an ending age constraint. It requires the average age of the forest at the end of the planning horizon to be at least $\frac{\text{Age}}{T}$ years, preventing the model from over-harvesting the forest. In the example below, the minimum average ending age was set at 40 years, or $\frac{1}{2}$ the optimal economic rotation.

Constraint sets (20)-(23) identify the stand prescription, mature patch size, and the $\Omega'_{pq}$ variables as binary.
Methods

This section describes how the two-objective ‘frontier’ methods, discussed in Tóth et al. (in press) were modified in this study to identify the set of efficient (or Pareto-optimal) solutions with respect to three or more objectives. Since both the Weighted Objective Functions method and the Tchebycheff-Metric-based method rely on the same algorithm (the Triangles Algorithm), they are discussed together. The mechanics of the Alpha-Delta and the $\varepsilon$-Constraining methods are also very similar.

Although Chalmet et al. (1983) proposed an algorithm that would extend their bi-objective Triangles Method (not to be confused with the Triangles Algorithm) to the general $n$-objective case, their approach is not evaluated in this study for the following reasons. Their proposed algorithm finds $|Z|$ efficient solutions with respect to $n$ objectives by solving at least $n|Z| + 1$ integer programs. This is not an efficient way to identify the efficient set given that the $\varepsilon$-Constraining method (Sadagopan et al. 1982), which has already been found computationally inefficient in the bi-objective case (Tóth et al., in press), finds the same number of solutions by solving only $n|Z|$ integer programs.

In the bi-criteria case, both the Alpha-Delta and the $\varepsilon$-Constraining methods find the Pareto-optimal solutions by progressively moving from one end of the efficient frontier to the other (Tóth et al., in press). The Alpha-Delta method uses a weighted objective
function where one objective has a disproportionately larger normalized weight than the other. The allocation of weights is constant throughout the algorithm. At each iteration, a new Pareto-optimal solution is found by optimizing the weighted objective function subject to a constraint that restricts the attainment on the objective with the lesser weight to be greater than or equal to (or less than or equal to when the objective is to be minimized) the attainment from the previous solution plus (or minus for minimization) a conveniently small \( \delta \). This constraint is not imposed on the problem at the first iteration since there is no previous solution available. The algorithm terminates when the attainment on the objective with the lesser weight is equal to its ideal value (i.e., the attainment value that is obtained by optimizing only this objective).

The mechanism of the \( \varepsilon \)-Constraining method is very similar. A key difference is that each new solution is found in two steps instead of one. First, one objective is optimized subject to a constraint that restricts the attainment on the second objective to be greater than (less than for minimization) or equal to the attainment from the previous solution plus (or minus) a conveniently small \( \delta \). Let \( \Phi \) denote the resulting objective function value. Next, the problem is turned around and the objective that was constrained in the first step is now optimized subject to a new constraint that restricts the objective that
was optimized in the first step to be equal to \( \Phi \). The second step is needed to ensure that a Pareto-optimal solution has been found. The stopping rule is the same as in the Alpha-Delta method.

An important common characteristic of the two methods is that as the solutions are progressively found along the efficient frontier, while moving from one end to the other, the attainment of one of objective function values gradually gets worse at each new solution while the attainment of the other objective function improves. In the case of the Alpha-Delta Method, the objective function value that is degraded at each step corresponds to the objective with the highest weight; in the case of the \( \varepsilon \)-Constraining method, it corresponds to the objective that is maximized first at each iteration. This algorithmic property of a ‘gradually degrading’ objective function value can be beneficial in decision making as it enables one to find efficient management alternatives that are very similar in terms of the achievements on the competing objectives.

The generalized \( n \)-objective versions of the two methods use the same idea of gradually degrading the attainment values for one objective. Strictly speaking, each new iteration of the algorithms generates an efficient solution (a management alternative) that
has a consistently worse attainment value on one of the objectives than the efficient solution from the previous iteration.

**The Alpha-Delta Method**

*Figure 5-1* illustrates the mechanics of the Alpha-Delta Method as applied to a tri-criteria problem. To simplify the exposition, so that all objectives are to be maximized, the third objective is expressed as maximizing the negative of the total amount of edge rather than minimizing the total edge. As in the bi-criteria case (Tóth et al. *in press*), a weighted objective function is used with a high weight assigned to one objective that is orders of magnitude higher than the weights assigned to the other two objectives. The weights are normalized using the “Best Value” approach as described in Tóth et al. *(in press)* and are fixed throughout the algorithm. In order to find the “best values,” each of the objectives must be optimized without regard to the other objectives at the initialization phase of the algorithm (*Figure 5-2*). The solution vector that consists of these optimal objective function values is called the “ideal solution.” It is ideal because these objective function values cannot be achieved simultaneously if the objectives themselves conflict with one another.
At the first iteration, the weighted objective function, which can be represented as a slightly tilted plane in the space of the achievable objective function values (Figure 5-1), is maximized. The relative difference in the weights assigned to the respective objectives is controlled by parameter $\alpha$, the slope of the tilted plane. This parameter has to be small enough to not miss any solutions but must be greater than zero to avoid finding dominated solutions.

To illustrate these undesirable situations, suppose the first Pareto-optimal solution to the tri-criteria problem is represented by Point (1) in the left part of Figure 5-1. If $\alpha$ is zero, there is a possibility that a dominated solution, for example Point (1'), would be found.
This solution provides exactly the same achievement on NPV as Point (1) but a lower achievement on both of the other two objectives. In other words, even though Point (1) represents a better alternative than Point (1′) in all respects, a setting of \( \alpha = 0 \) would not guarantee that Point (1) is identified rather than Point (1′). On the other hand, if \( \alpha \) is not small enough, there is a chance that Point (1) would be missed entirely, since a more tilted plane would pick up Point (1″) first and miss Point (1). Although Point (1″) is also an efficient solution, it should be found only after Point (1) is found to maintain the pattern of gradually decreasing NPVs.

Now, suppose that Point (1) is found after the first iteration with achievements \( N_1 \), \( H_1 \), and \( E_1 \) on the three competing objectives. As long as \( \alpha > 0 \), the following will be true for the rest of the efficient solutions: \( NPV < N_1 \) and either \( HABITAT > H_1 \) or \( -EDGE > E_1 \).

The latter two of these constraints can be used to restrict the search for the remaining solutions to a search space whose vertical projection is represented by the grey area in the left part of Figure 5-1. Inequality \( NPV < N_1 \) must hold for any solution in this region, because if \( NPV \geq N_1 \) was true for any one of the remaining efficient solutions then that solution would have been found at the first iteration instead of Point (1). As long as \( \alpha \) is small enough, we can rule out the region where \( HABITAT \leq H_1 \) and \( -EDGE \leq E_1 \) are both
true since if there were a solution in that region that dominates Point (1) it would have to have a higher $NPV$ than $N_I$ and should have been found in the previous iteration. In order to ensure that the search space is confined to $HABITAT > H_I$ or -$EDGE > E_I$ at the second iteration, the following set of constraints must be added to the original problem (see the `Model Formulation` section for the original problem).

\[
\lambda \geq (H_I + \delta_{hab})y_1 \tag{24}
\]

\[
- \sum_{t \in T} \mu_t \geq (E_I + \delta_{edge})y_2 \tag{25}
\]

\[
y_1 + y_2 = 1 \tag{26}
\]

\[
y_1, y_2 \in \{0, 1\} \tag{27}
\]

where $\lambda$ = the minimum area of mature forest habitat in patches over all periods;

$\mu_t$ = the total perimeter of mature forest habitat patches in period $t$;

$H_I$ = achievement on $\lambda$ from iteration one;

$E_I$ = achievement on $-\sum_{t \in T} \mu_t$ from iteration one;

$\delta_{hab}, \delta_{edge}$ = user-defined, sufficiently small constants;

$y_1, y_2$ = binary variables that ensure that only one of the constraints (24) and (25) is enforced.
Constraints (24)-(27) ensure that either the minimum area of mature forest habitat patches over all periods ($\lambda$) is strictly greater than $H_1$ or the total perimeter of the patches \( \sum_{t \in T} \mu_t \) is strictly smaller than $E_1$. The strictly greater (or smaller) requirement is needed to avoid repeatedly picking up the same solution, Point (1) in this case, or a solution dominated by Point (1). This requirement is achieved by adding sufficiently small constants, $\delta_{\text{hab}}$ and $\delta_{\text{edge}}$ to the bounds on habitat area ($H_1$) and perimeter ($-E_1$). The either-or relationship between constraints (24) and (25) is achieved by using constraints (26)-(27), which require that either $y_1 = 1$ and $y_2 = 0$ or vice versa. If $y_1 = 1$ then only constraint (24) is enforced, and if $y_2 = 1$ then only constraint (25) is enforced.

The problem is then re-solved with these constraints. Suppose Point (2) is found in this second iteration, with achievements $N_2$, $H_2$, and $E_2$ on the three objectives (Figure 5-1, right). Now, any remaining efficient solutions must be found within an objective space whose vertical projection can be represented by the grey area in Figure 5-1 (right). This additional restriction of the search space is achieved by adding the following constraints to the problem (the original problem with constraints (24)-(27)).

\[
\lambda \geq (H_2 + \delta_{\text{hab}})y_3 \\
-\sum_{t \in T} \mu_t \geq (E_2 + \delta_{\text{edge}})y_4
\]
\begin{align*}
y_3 + y_4 &= 1 \\
y_3, y_4 &\in \{0, 1\}
\end{align*}

These constraints work much like constraints (24)-(27). The additional new bounds on the minimum habitat area and the negative of the total perimeter are \(H_2 + \delta_{\text{hab}}\) and \(E_2 + \delta_{\text{edge}}\), respectively. After each iteration, a new quadruplet of constraints like (24)-(27) or (28)-(31) is added to the problem. The process is repeated until the problem becomes infeasible.

Figure 5-2 illustrates the general implementation of this algorithm when applied to \(n\)-objective problems. In the first step, the ideal solution is identified which is needed to scale and normalize the weights assigned to the respective objectives. In Step 2, a weighted objective function is generated with a very high weight assigned to one objective and negligible weights assigned to the other objectives. This weighted objective function is then maximized subject to the original set of constraints \(x \in X\). If this problem is infeasible than stop; there are no solutions to the problem. If the problem is feasible, then solve it and use the attainment values on the objectives with the smaller weights \(F_i^k\), for \(i \in P \setminus \{j\}\) to build and add a new set of constraints to the original problem (Step 3 and 4). These constraints will be similar to (24)-(27) with the only difference that there are \((n-1)\)
restricted objectives now instead of two. Using the same ‘either-or’ integer programming technique as in (24)-(27), only one of the \( n \) constraints \( f_i(x) \geq F^k_i + \delta_i \) (for \( i = 1, \ldots, n \) but \( i \neq j \)) must hold.

**Step 1: Obtain Ideal Solution**

Set \( k = 1 \)

**Step 2:** Solve \( P_{AD} = \text{Max} \left\{ \sum_{i \in P \setminus \{j\}} \lambda_i f_i(x) : \sum_{i = 1}^{P} \lambda_i = 1, \lambda_i \geq 0, x \in X \right\} \)

where \( \lambda_j > \lambda_i \) (\( j \) is fixed, \( i \in P \setminus \{j\} \)) are fixed normalized weights, \( f_i(x) \) and \( f_j(x) \) are objective functions and \( X \) is the feasible set.

**Step 3:** Call this solution: \( \text{EFS}(k) = (F^k_1, \ldots, F^k_i, \ldots, F^k_P) \) where \( F^k_i \) are the attainments values on objectives \( i \in P \setminus \{j\} \).

**Step 4:** Add the following constraints to feasible set \( X \):

\[
\begin{align*}
 f_i(x) \geq (F^k_i + \delta_i) y^i_i \quad &\text{for } i = 1, \ldots, P \\
 \sum_{i \in P} y^i_i = 1 \\
 y^i_i \in \{0, 1\}
\end{align*}
\]

where \( \delta_i \) are sufficiently small constants for \( \forall i \in P \)

**Step 5 (Optional):**

 Eliminate redundant constraints from \( X \)

Figure 5-2. The Alpha-Delta Algorithm for \( n = |P| \) objectives
Step 5 checks whether the newly constructed constraint set from Step 4 dominates any of the previously constructed sets. If it does, i.e., if $F_i^m \leq F_i^k$ (for each $i \in P \setminus \{j\}$ and for $m < k$) then the dominated constraint set (the one that was generated in Iteration $m$) can be eliminated from the problem. This step is optional because, as our preliminary runs indicate, removing the redundant constraint sets this way does not necessarily result in reduced solution times. Furthermore, depending on the integer programming solver used, these constraints will probably be eliminated automatically and efficiently in the solver’s preprocessing phase.

After removing the redundant constraints, the new integer program is solved and the process (Steps 2-5) is repeated until the problem becomes infeasible.

The $\varepsilon$-Constraining Method

The $\varepsilon$-Constraining Method is very similar to the Alpha-Delta Method. The key difference is that at each iteration $n$ integer programs are solved, as opposed to one (for an $n$-objective problem), to guarantee an efficient solution. Suppose the following iteration is the $k$-th iteration. First, one of the $n$ objectives, say $f_1^k(x)$ is maximized without regard to the rest of the objectives. If this problem is infeasible, the algorithm terminates; no more
efficient solutions exist. Otherwise, the problem is solved and the resulting objective function value, $F_1^k$ is recorded. Next, another objective is maximized, say $f_2^k(x)$ subject to $f_1^k(x) \geq F_1^k$. This problem is feasible since we know there exists at least one solution with $f_1^k(x) = F_1^k$. Call the objective function value of the resulting solution $F_2^k$. Now, a third objective is maximized subject to $f_1^k(x) \geq F_1^k$ and also to $f_2^k(x) \geq F_2^k$. The process is repeated until each of the objectives is maximized (Step 1 - 2 in Figure 5-3).

When the last objective, $f_n^k(x)$, is maximized, the rest of the objectives are constrained to $f_1^k(x) \geq F_1^k$, $f_2^k(x) \geq F_2^k$, ..., $f_{n-1}^k(x) \geq F_{n-1}^k$, where $F_i^k$ ($i = 1, 2, ..., n-1$) is the objective function value that was obtained by maximizing $f_i^k(x)$. The resulting objective function value, $F_n^k$, together with the previously obtained $F_1^k, F_2^k, ..., F_{n-1}^k$ constitute the attainment values on the $n$ objectives for efficient solution $k$ (Step 3). Steps 4 and 5, as well as the stopping rule, are exactly the same as in the Alpha-Delta Algorithm.

Figure 5-4 illustrates the first two iterations of the ε-Constraining Algorithm as applied to the tri-criteria forest planning problem presented earlier. At the first iteration, the NPV is maximized first without regard to the rest of the objectives. Call the resulting objective function value MNPV. This step is illustrated in Figure 5-4 (left) with a grey arrow numbered ‘1’. Second, the HABITAT is maximized subject to NPV $\geq$ MNPV.
Initialization: Set $i = k = 1$

Step 1: Solve

$$P_{\text{CON}} = \max \left\{ f_i(x) : x \in X \right\}$$

where $f_i(x)$ is objective function $i \in P$ and $X$ is the feasible set.

Step 2:
- Call attainment value on objective $i$: $F^i_k$
- Add constraint $f_i(x) \geq F^i_k$ to feasible set $X$.

Step 3: Call this solution:

$EFS(k) = (F^1_k, \ldots, F^p_k)$ where $F^i_k$ are the attainments values on objectives $i \in P$.

Step 4: Add the following constraints to feasible set $X$:

$$f_i(x) \geq (F^i_k + \delta_i) y^i_k$$

$$\sum_{i \in P} y^i_k = 1$$

$$y^i_k \in \{0, 1\}$$

where $\delta_i$ are sufficiently small constants for $\forall i \in P$.

Step 5 (Optional): Eliminate redundant constraints from $X$.

STOP

Figure 5-3. The $\varepsilon$-Constraining Algorithm for $P$ objectives

(Arrow ‘2’ in Figure 5-4). Call the resulting objective function value $H_1$. Third, the –

EDGE is maximized (or, equivalently, EDGE is minimized) subject to both $\text{NPV} \geq \text{MNPV}$ and $\text{HABITAT} \geq H_1$ (Arrow ‘3’ in Figure 5-4). The resulting objective function value, $E_1$, together with MNPV and $H_1$ are the attainment values on the three objectives for the first
efficient solution. This solution is represented as Point (1) in Figure 5-4 (left). Using $E_1$ and $H_1$, a set of ‘either-or’ constraints, identical to (24)-(27), is built and added to the original problem. This completes the first iteration.

Iteration 2 follows the same steps as Iteration 1, except that the problem now is constrained by (24)-(27). Suppose the attained objective function values are $N_2$, $H_2$ and $E_2$ on the NPV, HABITAT and –EDGE, respectively (Figure 5-4, right). Using now $E_2$ and $H_2$, another set of ‘either-or’ constraints, identical to (28)-(31), is built and added to the problem. This completes the second iteration. The iterations are repeated until the problem becomes infeasible.

![Figure 5-4. The first two iterations of the ε-Constraining Algorithm with 3 objectives](image)

190
The Weighted Objective Functions and the Tchebycheff-Metric based Methods

A common characteristic of the Weighted Objective Functions and the Tchebycheff Metric-based methods is that both approaches make use of an efficient decomposition of weights when applied to bi-objective problems (Tóth et al. in press). In the case of the Weighted Objective Functions Method, these weights are assigned to the competing objectives and the sum of these weighted objectives is maximized. In the case of the Tchebycheff Metric-based approach, the weights are assigned to the components of the Tchebycheff Metric, which measures the maximum difference between the attainment values of a potential solution and that of the ideal solution (Tóth et al. in press). The weighted Tchebycheff Metric is then minimized to obtain solutions that are as close the ideal solution as possible. Although varying the relative weights on the competing objectives or the components of the Tchebycheff Metric will often yield different efficient solutions, it is also possible that two different combinations of weights result in the same solution. To minimize the number of redundant solutions and the amount of computer time that is needed to find these solutions, Tóth et al. (in press) used an algorithm that decomposes the set of possible normalized weight combinations into sections (line segments in the bi-criteria case) that correspond to the same efficient solutions (Eswaran et al. 1989). The decomposition for bi-objective problems is based on the fact that if two
different weight combinations yield the same solution then any linear combination of these weights will do so as well. Thus, these linear combinations of weights can be eliminated from further consideration.

The decomposition of the weight space is not as straightforward if there are three or more objectives. If there are only three objectives, the set of possible normalized weight combinations can be mapped as a triangle (Figure 5-5). Since the weights are normalized, i.e., they sum up to one, two of the three weights that are assigned to the competing objectives or the components of the Tchebycheff Metric determine the third weight. This is why the set of weights forms a triangle for tri-criteria problems. The apexes of the triangle represent the weight combinations when a weight of one is assigned to one objective (or one component of the Tchebycheff Metric) and zeros are assigned to the other two. This triangle is illustrated in Figure 5-5 with apexes (1,0,0), (0,1,0) and (0,0,1). The proposed procedure, the Triangles Algorithm, decomposes this triangle into triangular sections that correspond to the same efficient solutions. At each iteration, one triangle is considered. If the three weight combinations that represent the three apexes of the triangle yield the same solution then no further decomposition of that triangle is necessary. Any point within the triangle (or, equivalently, any linear combination of the weights at the apexes) will yield the
same solution. If, however, the weights at the apexes yield two or three different solutions, the triangle must be divided into four smaller but identically shaped triangles. These are

Figure 5-5. The Weight Triangle

sub-triangles\( ((1/2, 1/2, 0), (0, 1/2, 1/2), (1/2, 0, 1/2)) \), \( ((1, 0, 0), (1/2, 1/2, 0), (1/2, 0, 1/2)) \), \( ((1/2, 1/2, 0), (0, 1/2, 1/2), (0, 1, 0)) \), and \( ((0, 0, 1), (0, 1/2, 1/2), (1/2, 0, 1/2)) \) in Figure 5-5. The apexes of the sub-triangles are either identical to one of the apexes of the “parent” triangle \((1,0,0), (0,1,0), (0,0,1)\) or are constructed as the mean of the two of those apexes. If two of the three solutions from the “parent” triangle was the same, e.g., weights \((1,0,0)\)

and \((0,1,0)\) both yielded the same solution, then weight combination \((1/2, 1/2, 0)\), which is a
linear combination of \((1,0,0)\) and \((0,1,0)\), can be assigned that solution as well. There is no need to solve the problem with weights \((1/2, 1/2, 0)\).

At the next iteration, one of the four sub-triangles is selected and the same process is followed as in the first iteration. The weights that define the apexes of the sub-triangle are applied to the objectives of the problem (or to the Tchebycheff Metric, depending on which method is used). It is entirely possible that one of the weight combinations corresponding to one of the apexes of the sub-triangle has already been applied to the problem and solved at a previous iteration, either as part of the larger problem or when an adjacent triangle was explored. In this case, there is no need to solve the problem with these weights again. The solution from the adjacent triangle can be imported and used in the comparisons that are needed to determine if the current sub-triangle must further be decomposed. Those of the three problems that have not been solved before or are not linear combinations of other weights that yield the same solution are therefore solved and compared as in the first iteration.

It is necessary to determine a minimum mesh size – i.e, a minimum size of triangle that will not be decomposed further. Thus, the algorithm terminates when there are no more sub-triangles left to decompose or the largest difference between the weight
combinations that correspond to the apexes of the remaining sub-triangles is smaller than this predefined limit.

The flowchart in Figure 5-6 illustrates the logical flow of the Triangles Algorithm. The following notation is used. Let $T$ be the set of “active” (unexplored) triangles. Let

\[
W_i = \begin{pmatrix}
  w_{i1}^i & w_{i2}^i & w_{i3}^i \\
  w_{i1}^i & w_{i2}^i & w_{i3}^i \\
  w_{i1}^i & w_{i2}^i & w_{i3}^i
\end{pmatrix}
\]

denote the weights associated with the apexes of triangle $\Delta_i \in T$, and $\Lambda_i = -\max \left( w_{i1}^i - w_{i2}^i, w_{i1}^i - w_{i3}^i \right)$ denote the depth of $\Delta_i$. This latter metric, ‘depth’ describes the size of a given triangle and is used in the algorithm to identify the largest triangles. The greater the value of $\Lambda_i$, the larger triangle $\Delta_i$ is. The algorithm decomposes the largest active triangles first. Lastly, let $F_i^k$ (for $k = 1, 2, 3$) denote the objective function values that correspond to the solutions of problems

\[
P_i^k = \max \left\{ w_{k1}^i f_1(x) + w_{k2}^i f_2(x) + w_{k3}^i f_3(x) : x \in X \right\}
\]

for $k = 1, \ldots, 3$, respectively, where $f_1(x), f_2(x)$, and $f_3(x)$ are the objective functions.

Step 1 and 2 is the initialization phase of the algorithm. Step 1 is to obtain the ideal solution, which is needed to scale and normalize the weights for both the Weighted Objective Functions and the Tchebycheff-Metric-based methods. A scalar $\lambda$ is also defined and is used to limit the size of the triangles that need to be decomposed. Scalar $\lambda$ is use-
defined. At this point, set \( T \) is empty. Step 2 is to add the first triangle to the list of active triangles (set \( T \)). This triangle is the so-called ‘parent’ triangle whose apexes represent weight combinations \((1,0,0)\), \((0,1,0)\) and \((0,0,1)\). The solutions to these single-objective problems have already obtained in Step1 when the ideal solution was identified.

At the beginning of each iteration, set \( T \) is checked. If set \( T \) is empty, the algorithm terminates. If set \( T \) is non-empty, then one of the largest triangles, say triangle \( \Delta_i \) is selected. If \( \Delta_i \) is smaller than the predefined \( \lambda \) or the solutions that correspond to the apexes of \( \Delta_i \) are identical, then \( \Delta_i \) is removed from set \( T \). Otherwise, four new triangles are created \(( \Delta_{\|1\|1}, \Delta_{\|1\|2}, \Delta_{\|1\|3}, \Delta_{\|1\|4} \)) with the following weights on the apexes (Step 3):

\[
\begin{align*}
W_{\|1\|1} &= \begin{bmatrix}
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33}) \\
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33})
\end{bmatrix}, \\
W_{\|1\|2} &= \begin{bmatrix}
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33}) \\
\frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{22} + w_{32}) & \frac{1}{2}(w_{23} + w_{33}) \\
\frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{22} + w_{32}) & \frac{1}{2}(w_{23} + w_{33})
\end{bmatrix}, \\
W_{\|1\|3} &= \begin{bmatrix}
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33}) \\
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33})
\end{bmatrix}, \\
W_{\|1\|4} &= \begin{bmatrix}
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33}) \\
\frac{1}{2}(w_{11} + w_{21}) & \frac{1}{2}(w_{12} + w_{22}) & \frac{1}{2}(w_{13} + w_{23}) \\
\frac{1}{2}(w_{11} + w_{31}) & \frac{1}{2}(w_{12} + w_{32}) & \frac{1}{2}(w_{13} + w_{33})
\end{bmatrix}
\end{align*}
\]

The next step is to generate and
Step 1: Obtain Ideal Solution
Set \( T = \emptyset \). Define \( \lambda \).

Step 2: Add first triangle \( \Delta_1 \), with weights 
\( W_1 = I \), to set \( T : T = T \cap \{ \Delta_1 \} \),
where \( I \) is the 3x3 identity matrix.

If \( T = \emptyset \),

Select a \( \Delta_i \in T \) with greatest depth \( \Lambda_i \).

Step 3:
Create 4 new triangles \( \Delta_{1/1}, \Delta_{1/2}, \Delta_{1/3}, \Delta_{1/4} \), with weights 
\( W_{1/1}, W_{1/2}, W_{1/3}, W_{1/4} \) and 
\( W_{1/4} \) assigned to the apexes. Set 
\( \Delta_{1/2} = F_i^2, \Delta_{1/3} = F_i^3 \), and 
\( \Delta_{1/4} = F_i^4 \).

Step 4:
No need to solve problems 
\( P_{1/1}^1, P_{1/2}^1, P_{1/3}^1, P_{1/4}^1 \):
\( F_{1/1}^1 = F_{1/2}^1 = F_{1/3}^1 = F_{1/4}^1 = F_i^1 \).

Solve only 
\( P_{1/1}^2, P_{1/2}^2, P_{1/3}^2, P_{1/4}^2 \):
\( F_{1/1}^2 = F_{1/2}^2 = F_{1/3}^2 = F_{1/4}^2 = F_i^2 \).

No need to solve problems 
\( P_{1/1}^3, P_{1/2}^3, P_{1/3}^3, P_{1/4}^3 \):
\( F_{1/1}^3 = F_{1/2}^3 = F_{1/3}^3 = F_{1/4}^3 = F_i^3 \).

Step 5: Add the 4 new triangles to set 
\( T : T = T \cap \{ \Delta_{1/1}, \Delta_{1/2}, \Delta_{1/3}, \Delta_{1/4} \} \).

No need to solve problems 
\( P_{2/1}^1, P_{2/2}^1, P_{2/3}^1, P_{2/4}^1 \): 
\( F_{2/1}^1 = F_{2/2}^1 = F_{2/3}^1 = F_{2/4}^1 = F_i^1 \).

Solve only 
\( P_{2/1}^2, P_{2/2}^2, P_{2/3}^2, P_{2/4}^2 \):
\( F_{2/1}^2 = F_{2/2}^2 = F_{2/3}^2 = F_{2/4}^2 = F_i^2 \).

No need to solve problems 
\( P_{2/1}^3, P_{2/2}^3, P_{2/3}^3, P_{2/4}^3 \):
\( F_{2/1}^3 = F_{2/2}^3 = F_{2/3}^3 = F_{2/4}^3 = F_i^3 \).

Solve only 
\( P_{2/1}^4, P_{2/2}^4, P_{2/3}^4, P_{2/4}^4 \):
\( F_{2/1}^4 = F_{2/2}^4 = F_{2/3}^4 = F_{2/4}^4 = F_i^4 \).

Figure 5-6. The Triangles Algorithm for 3 objectives
solve the 12 problems \((P_{v|1}, P_{v|2}, P_{v|3}, \text{ and } P_{v|4})\) for \(i = 1, 2, 3\) with the weight combinations that correspond to the apexes of the four triangles (Step 4). At most, only three of the 12 problems would have to be solved, because the same weight combinations are assigned to more than one apex (Figure 5-5). Furthermore, not every one of these three problems might have to be solved if one of the three conditions in Step 4 hold (Figure 5-6), or they have been already solved in a previous iteration. Finally, the four new triangles are added to set \(T\) (Step 5) and the process starts all over again by selecting another triangle.

**A case study**

In order to illustrate and test the performance of the four methods discussed above in generating the trade-offs with respect to three objectives, an example hypothetical forest was created. This forest consisted of 50 stands and could be considered slightly over-mature, since approximately 40% of the area is between 60-100 years old and the optimal rotation is 80 years. The average stand size was 18 ha, and the total forest area was 900 ha. A 60-year planning horizon was considered, composed of three 20-yr periods. The four possible prescriptions for a given stand were to harvest the management unit in period 1, period 2, or period 3, or not at all. The minimum rotation age was 60 years. A maximum
harvest opening size of 40 ha was imposed, and groups of contiguous stands were allowed to be harvested concurrently as long as their combined area was less than the maximum opening size. All management units are smaller than the maximum harvest opening size. The wildlife species under consideration is assumed to need habitat patches that are at least 50 ha in size and at least 60 years old. Since the minimum habitat patch size is greater than the maximum harvest opening size, these patches must be composed of more than one management unit.

We applied the algorithms introduced in the Methods section to the tri-criteria integer program described in the Model Formulation section using CPLEX 9.0 (ILOG CPLEX 2003) on a dual processor Intel® XEON™ CPU 3.06 GHz computer with 3.0 GB RAM under a Windows platform (Microsoft Windows XP Professional Version 2002, Service Pack 1). Programs to automate the algorithms were written in Microsoft Visual Basic 6 using the ILOG CPLEX Callable Libraries. The relative MIP gap tolerance parameter (optimality gap) was set to $1.e-05$ (0.001%) (except the Tchebycheff-Metric-based Method where it was set to 0.0005%), the integrality tolerance parameter was set to $1.e-07$ (0.00001%), the MIP variable selection strategy parameter was set to ‘3’, and the MIP parallel threads parameter was set to “2” (i.e., parallel strong branching). The default
128KB working memory limit was used. The parameters of the Alpha-Delta and the $\epsilon$-Constraining algorithms, $\alpha$, $\delta_{ha}$, and $\delta_{edge}$ were set to $1^\circ$, 0.01 ha, and 0.47 m, respectively ($\alpha$ only applies to the Alpha-Delta Method). The depth parameter in the Triangles Algorithm ($\lambda$) was set to zero for both the Weighted Objective Functions and the Tchebycheff-Metric-based approaches. The reasons for choosing the above settings will be discussed in the next section.

The experiment addressed the following questions: (1) How many of the efficient solutions can each algorithm identify within three weeks of computer time? (2) How evenly are these solutions distributed along the efficient frontier? (3) How easily can the user of the methods filter the solutions in line with the DMs interests? (4) How easily do the methods generalize to the general $n$-objective case? and (5) How good are these solutions in terms of optimality? The fifth question refers to the fact that even though the optimality gap was set to very small values (smaller than the default values for CPLEX) for each algorithm, some methods might consistently generate solutions that are better than the ones generated by the other methods that are still within this range.
Results and Discussion

The Weighted Objective Functions Method found the highest number of Pareto-optimal management alternatives (n=112) within the preset time interval of three weeks. The Alpha-Delta method found 92, the Tschebyseff Method found 35, and the \( \varepsilon \)-Constraining method found 16 solutions. Figure 5-7 and 5-8 graph the solutions identified by each method in the objective function value space.

These results suggest that the \( \varepsilon \)-Constraining method is computationally too expensive to be useful. Additionally, the Alpha-Delta and the \( \varepsilon \)-Constraining methods found solutions mostly on one side of the efficient frontier, while the solutions identified by the other two methods were more evenly distributed. Even though the Weighted Objective Functions method identified more solutions overall and provided a better estimate of the entire efficient frontier, the Alpha-Delta method described one part of the frontier in more detail. The main reason for the difference between the two methods is that, unlike the Weighted Objective Functions method, the Alpha-Delta method is capable of identifying non-supported Pareto-optimas (Tóth et al. in press) so it can explore the efficient frontier in more detail than the Weighted Objective Functions method. However, as this method
Figure 5-7. Efficient alternatives found by the $\varepsilon$-Constraining- and the Alpha-Delta methods
Figure 5-8. Efficient alternatives found by the Weighted- and Tchebycheff methods
explores the efficient frontier starting from one end (from the highest levels of NPVs) and as it requires a set of either-or constraints and a set of corresponding binary variables to be added to the problem at each iteration, the integer program that needs to be solved becomes increasingly intractable as the algorithm proceeds. The structure of the tri-criteria harvest scheduling problem used in this experiment might also account for this increasing combinatorial complexity. The more NPV is forgone, i.e., the further the algorithm gets from the solutions with the highest NPVs, the more mature forest habitat can be preserved and thus the more harvest scheduling combinations exist that might have to be evaluated in the branch-and-bound process to find the optimal solution. Depending on the point where the integer program becomes intractable and what time constraints are imposed on the solution process, the Alpha-Delta method might or might not be able to scan the entire efficient frontier. One way to mitigate this problem is to run three separate algorithms each starting from a different end of the efficient frontier. The Alpha-Delta Algorithm can be instructed to work its way off the highest possible level of minimum habitat area or from the lowest possible length of perimeter. This way, only the central part of the efficient frontier would have to be explored using a larger burden of ‘either-or’ constraints, and the peripheral solutions could be found relatively easily. This combined algorithm can
terminate once the three sub-algorithms (or $n$ sub-algorithms for an $n$-objective problem) “meet” somewhere in the middle of the efficient frontier – for example, when they find an identical efficient solution. Of course, there is no guarantee that the integer programs that are progressively solved would not become intractable before the sub-algorithms meet. This combined approach, however, should increase the number of efficient solutions identified.

The Weighted Objective Functions method performed surprisingly well. It found the highest number of efficient solutions (despite the fact that this method cannot identify non-supported Pareto-optima) and the solutions were more evenly distributed than the ones found by the other methods. This result is in contrast with the Weighted Method’s performance when it was applied to a bi-criteria harvest scheduling problem. Tóth et al. (in press) reported that the Weighted Method found only 6 efficient alternatives for the test problem as opposed to the 36 found by the Alpha-Delta Method. The reason for the better performance of the tri-criteria Weighted Method might be that the integer programs that need to be solved at each iteration with this method are easier to solve than those of the Alpha-Delta Method. The Weighted Method does not require additional constraints and variables to be added to the problem, only the weights assigned to the objectives change.
This is a significant advantage. In the bi-criteria case, however, where the integer programs in the Alpha-Delta Method were not encumbered by new constraints and new variables at each iteration, the Weighted Method was not competitive (Tóth et al. in press).

Others have also pointed out that the Weighted Method might be a preferred alternative to solve multiple-objective integer programs if the original problem has a special structure that would be destroyed by using other methods (e.g., ReVelle 1993). This important observation is particularly useful if the integer program has a totally unimodular structure which allows integral solutions even if the integrality constraints are relaxed (Wolsey 1998). In these cases, destroying this structure in order to find the non-supported Pareto-optimal solutions, and thus better describing the efficient frontier, might not be worthwhile.

The Tchebycheff-Metric-based approach found only 35 solutions. Although these solutions were found relatively quickly (within a few days), the last iteration of the algorithm did not yield a solution within the pre-specified gap for 17 days. This result suggests that a time-limit on each iteration might help in finding more efficient solutions instead of wasting time on a single particularly hard problem. The problem with this solution, however, is that such a time limit would prevent the branch and bound algorithm
from guaranteeing the solutions at some corners are within the pre-specified optimality gap.

This would tend to increase the severity of the numerical problems discussed later in this section.

Table 5-1. Sub-optimal / dominated solutions found by the Tchebycheff method

<table>
<thead>
<tr>
<th>No.</th>
<th>NPV ($)</th>
<th>Habitat (ha)</th>
<th>Perimeter (m)</th>
<th>NPV ($)</th>
<th>Habitat (ha)</th>
<th>Perimeter (m)</th>
<th>in NPV (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2,425,602</td>
<td>50.0122</td>
<td>13,232</td>
<td>2,420,906</td>
<td>50.0122</td>
<td>13,232</td>
<td>0.1936%</td>
</tr>
<tr>
<td>2</td>
<td>2,418,969</td>
<td>50.0122</td>
<td>12,617</td>
<td>2,413,826</td>
<td>50.0122</td>
<td>12,617</td>
<td>0.2126%</td>
</tr>
<tr>
<td>3</td>
<td>2,415,660</td>
<td>50.0122</td>
<td>12,494</td>
<td>2,405,638</td>
<td>50.0122</td>
<td>12,494</td>
<td>0.4149%</td>
</tr>
<tr>
<td>4</td>
<td>2,394,599</td>
<td>54.1967</td>
<td>11,717</td>
<td>2,394,402</td>
<td>50.0122</td>
<td>11,717</td>
<td>0.0082%</td>
</tr>
<tr>
<td>5</td>
<td>2,370,737</td>
<td>54.1967</td>
<td>11,504</td>
<td>2,368,450</td>
<td>54.1967</td>
<td>11,504</td>
<td>0.0965%</td>
</tr>
<tr>
<td>6</td>
<td>2,259,221</td>
<td>50.8986</td>
<td>10,414</td>
<td>2,227,323</td>
<td>50.8986</td>
<td>10,414</td>
<td>1.4119%</td>
</tr>
<tr>
<td>7</td>
<td>2,244,630</td>
<td>152.7238</td>
<td>30,048</td>
<td>2,244,586</td>
<td>152.7238</td>
<td>30,048</td>
<td>0.0020%</td>
</tr>
<tr>
<td>8</td>
<td>2,147,230</td>
<td>169.0498</td>
<td>27,314</td>
<td>2,112,338</td>
<td>169.0498</td>
<td>27,314</td>
<td>1.6250%</td>
</tr>
<tr>
<td>9</td>
<td>1,999,368</td>
<td>167.477</td>
<td>23,184</td>
<td>1,959,335</td>
<td>167.477</td>
<td>23,184</td>
<td>2.0023%</td>
</tr>
</tbody>
</table>

In addition, nine of the solutions identified by the Tchebycheff Method were dominated by solutions found by the other methods. In these solutions, the attainment values on the NPV were significantly lower than in the solutions found by the other methods (Table 5-1). Why did the Tchebycheff Method find sub-optimal solutions even though the optimality tolerance gap was set the same as in the other methods? The answer lies in the structure of the Tchebycheff-Metric itself. As described earlier, this metric measures the maximum (weighted) difference in the attainments on the respective
objectives between two solutions, one of which is the ideal solution. These differences –
there is one for each objective function – are the components of the metric. The
Tchebycheff approach, described in this paper, minimizes the Tchebycheff-Metric and
finds efficient solutions by varying the weights on the components of the metric (Tóth et al.
in press). Once the maximum of these weighted components is minimized, there is no
incentive to further reduce the value of the other components. This is why the attainment
values on one objective (the NPV of the forest in this case) are sub-optimal. This is a
serious disadvantage of using the Tchebycheff-Metric in multiple-objective mathematical
programming. This problem, however, can be easily overcome by using another metric.
The $L_1$ Metric, for example, which measures the sum (instead of the maximum) of the
differences in the attainments on the respective objectives and also leads to a linear
objective function, does not suffer from this drawback.

In conclusion, as in the bi-criteria case, a combined approach of the Weighted and
the Alpha-Delta methods can be recommended. The forest planner could use the Weighted
Method to obtain a well-distributed subset of the efficient set and these results can be
presented to the DM. Then, in line with the DM’s interests, one area of the efficient
frontier could further be explored using the Alpha-Delta Method. This approach would take advantage of the benefits of both methods.

Besides using the Weighted Method as an initial filter, there are several other, indirect ways to control the ‘spacing’ of the efficient solutions along the efficient frontier with the proposed algorithms. Filtering the efficient set might be advantageous for a couple of reasons: (1) it can significantly reduce the computational burden and therefore the solution time, and (2) a well-distributed subset of the efficient frontier might provide a sufficient pool of alternatives for the DM to choose from or to focus the search to a sub-region of his or her interest along the efficient frontier.

Increasing the values of parameters $\alpha$, $\delta_{\text{hab}}$, and $\delta_{\text{edge}}$ in the Alpha-Delta- and parameters $\delta_{\text{hab}}$ and $\delta_{\text{edge}}$ in the $\varepsilon$-Constraining Algorithm will increase the spacing of the efficient solutions in the objective space. The settings of $\delta_{\text{hab}}$ and $\delta_{\text{edge}}$ will ensure that no two solutions will be found whose achievements on the minimum habitat area and the length of edge are both within a $\delta_{\text{hab}}$ and $\delta_{\text{edge}}$ radius, respectively. One would have to keep in mind, however, that the setting of these parameters cannot guarantee that a solution will be found right outside of these ranges since the distribution of the complete set of efficient solutions along the efficient frontier might be rather uneven (e.g., see Figure 5-7, 5-8).
The spacing of the efficient solutions with the Weighted Objective Functions and the Tchebycheff-Metric-based approaches can be controlled to some extent by adjusting parameter $\lambda$ in the Triangles Algorithm. This parameter, by defining the size of the sub-triangles that need to be explored, can limit the extent to which the weight space is decomposed. The greater the value of $\lambda$, the sooner the algorithm will reach the desired level of resolution and the smaller will be the number of efficient solutions that are found. The setting of $\lambda$ can increase the likelihood that solutions with very similar objective function achievements will be avoided, but this cannot be guaranteed because very different weight combinations can lead to similar or even identical solutions. Thus, the Weighted Objective Functions and the Tchebycheff-Metric-based approaches can only indirectly, and without any certainty, control the spacing of the efficient solutions.

One additional advantage of the Alpha-Delta- and the $\varepsilon$-Constraining methods over the other two approaches is that their generalization to the $n$-objective case is fairly straightforward – at least, from a technical, integer programming point of view. Figures 5-2 and 5-3 illustrate the logical flow of these algorithms as applied to this general case. Unfortunately, as modeling more objectives requires more ‘either-or’ constraints and variables to be added to the problem at each iteration, the accumulation of these extra
constraints tend to make the problem increasingly intractable as the algorithms progress.

Starting the algorithms from each end of the efficient frontier, as explained earlier, might enable the algorithms to find more efficient solutions before the problems become intractable.

Generalizing the Triangles Algorithm to the $n$-objective case is not as straightforward as with the above two methods. Instead of the triangles of the tri-objective case, $n$-dimensional polyhedra (tetrahedra in the 4-objective case) would have to be decomposed into $n$-dimensional sub-polyhedra. Since each $n$-dimensional polyhedron has $n$ apexes, $n$ new problems would have to be solved and compared ($n!$ pair-wise comparisons) at each iteration. This increased computational burden, however, might be offset by the fact that the IP sub-problems (the problems that are solved at the apexes of the $n$-dimensional polyhedra) are simpler than those of the Alpha-Delta- or the $\varepsilon$-Constraining methods. Unlike the feasible region of the latter two methods, the feasible region of the IP sub-problems in the Triangles Algorithm is constant, only the weights on the objectives (or the components of the Tchebycheff-Metric) change. As the tri-criteria example that was discussed earlier demonstrates, this can be a great computational advantage over the Alpha-Delta or the $\varepsilon$-Constraining methods.
Finally, two important numerical issues need to be addressed that have been uncovered in the course of this study. One is related to the integrality tolerance setting and the other is to the optimality tolerance setting.

Due to the computational limitations of the current software and hardware technologies, integer programming solvers cannot always guarantee that the 0-1 variables used in these models take values exactly equal to one or zero in a feasible solution. Depending on the setting of the integrality tolerance parameter, which defines the maximum allowable deviation from the value of 1 or 0, 0-1 variables might take values such as $9.06 \times 10^{-6}$ or 0.99998. The default setting of this parameter in CPLEX 9.0 is $1.06 \times 10^{-5}$ and would allow these values. Although in most cases this setting ensures a sufficiently accurate numerical representation of a 0-1 program, ‘either-or’ constraints such as (24)-(27) might malfunction if parameters $\delta_{\text{hab}}$ or $\delta_{\text{edge}}$ are set too low relative to the integrality tolerance parameter. The minimum amount of mature forest habitat ($\lambda$) in constraint (24), for example, can potentially take a value that is substantially less than 100 ha even though $H_1$ is set to 100 ha, if $\delta_{\text{hab}} = 0.001$ and $\gamma_1 = 0.99998$ with a default integrality parameter. This is why a setting of $\alpha = 0.01$, $\delta_{\text{hab}} = 0.01$ ha, and $\delta_{\text{edge}} = 0.47$ m was used with an
integral tolerance parameter of $1.e-07$ for the Alpha-Delta- and $\varepsilon$-Constraining methods that utilize the ‘either-or’ constraint structure.

Another problem is that large-scale integer programming problems can rarely be solved to full optimality. Thus, an optimality tolerance parameter is used that instructs the IP solver to terminate once the percentage deviation between the objective function values of the current best integer solution and the dual bound falls below a pre-specified limit. This is a convenient tool of exact optimization because the user can control the desired optimality of the solutions given computer time constraints (McDill and Braze 2001).

However, sub-optimal solutions can create numerical problems in solving multiple-objective IPs.

One obvious direct consequence of sub-optimality is that the multi-objective techniques proposed in this paper might produce dominated solutions. Furthermore, an alternative that is found in one iteration of one of the algorithms and that is supposed to be Pareto-optimal might be dominated by a solution found in another iteration. The only way to avoid this situation is to set the optimality tolerance gap as small as possible.

Another critical issue related to sub-optimal solutions applies to the Triangles Algorithm. A critical step in the Triangles Algorithm is to check if the solutions that
correspond to the apexes of a triangle are the same. If these solutions are sub-optimal, it is possible that the triangle itself actually encompasses a set of weight combinations that should yield the same efficient solution but the solutions found with the weights at the apexes are different. A solution at one of the apexes can potentially dominate the solutions at the other apexes. Furthermore, two of the three solutions at the apexes can both dominate the third solution. Unfortunately, there is no sure way to tell for a weight combination, that leads to a sub-optimal solution, what true optimal solution it corresponds to even if there are a number of known solutions that dominate this sub-optimal solution.

Figure 5-9 illustrates this situation.

![Figure 5-9. Sub-optimality and multiple-dominance](image-url)
Suppose Points A, B, C, D and F represent the solutions that have already been found. The attainment on the third objective (edge length) is assumed to be the same for all of these solutions for the sake of simplicity. Point F is clearly dominated by Points A, B and C (i.e., Point F is a sub-optimal solution). The problem is that it is possible that none of these three solutions is the real optimal solution for the weights that lead to the sub-optimal solution of F. Suppose that Points F, B and C are solutions that correspond to the apexes of a sub-triangle in the Triangles Algorithm (Figure 5-9, upper right). Although it might seem reasonable to assign either solution B or C, which both dominate solution F, to the apex where solution F was found, given they would both yield a higher objective function value than Point F, such a decision would be arbitrary, and it is even possible that a new efficient solution (such as Point E) might have been missed. If solution B were assigned to the weight combination that yielded solution F, the weight space between the weights that yielded solution F and the weights that yielded solution B would not be decomposed further in the Triangles Algorithm. To avoid missing efficient solutions this way, a sub-triangle was always decomposed if the solutions that correspond to its apexes are different regardless of whether they dominate each other or not. At the end of the
algorithm, however, all dominated solutions were discarded, as new solutions would not be found anymore.

After a good pool of efficient solutions have been identified using one of the methods discussed above (or a combination of them), the next question is how to present the alternatives to the decision makers. Effectively visualizing the trade-offs between the alternatives is a challenging task, especially if three or more competing objectives are present. Using 3D graphs to map the attainment values of each alternative with respect to three objectives is one solution (e.g., Figure 5-7 and 5-8). The attainment values on a fourth objective can be represented by color codes or by changing the size of the dots that represent the solutions. 3D graphs are particularly powerful if the images can be rotated and zoomed in and out by the users (the decision makers). Another way to visualize the solutions is the “book” method where each page is a 2D projection of the attainment values of the solutions with respect to three objectives. The number of solutions that are represented on any one page is limited. Only those alternatives whose attainments on one objective (the objective whose axis is perpendicular to the plane of the projection) fall within a small range of values are included in a page. For example, Figure 5-8 could be represented as a “book” of 2D efficient frontiers (with respect to the NPV and the length of
Effectively visualizing the trade-offs with respect to four or more objectives is a very challenging task. One way to approach this task is to first generate efficient frontiers with respect to the two or three most important or most severely conflicting objectives, or with respect to the objectives that are quantifiable. The decision makers can then pick the best compromise solutions from clusters of alternatives (alternatives that are very similar in terms of the achievements on the objectives in question) based on a third or fourth objective that was not explicitly accounted for in the model. This objective might not be as important as the ones in the model or it might not be quantifiable.

**Conclusions**

This article presented four ways to evaluate the trade-offs, or equivalently, to generate a set of Pareto-optimal solutions to integer programs with three or more competing objectives. The results from one test of the different algorithms suggest that a combined utilization of the beneficial properties of both the Weighted Objective Functions and the Alpha-Delta methods would probably work the best in a real decision making situation.
Identifying the best ways to visualize the pros and cons of the efficient management alternatives and optimizing the interaction with the decision makers are key issues that need to be addressed in future research in order to successfully apply the proposed methods.

These approaches are computationally very expensive. At this point, they can only be applied to small-scale forest planning problems on a pilot-study basis. These pilot studies, however, are very important. Employing real decision makers or interested citizen groups, these studies could answer questions such as (1) how important is it to find the non-supported Pareto-optima, (2) how to visualize and present the alternatives to the DMs, and (3) to what extent can these methods promote consensus between multiple stakeholders. By the time these questions are answered, our computational capabilities might improve to such a degree that would allow us to solve realistically large problems.

The primary value of generating the set of efficient solutions, however, is to help the decision makers acquire a more holistic understanding of the problem by providing information about the trade-offs, the production possibilities and the degree of incompatibility between the competing objectives. This in turn can facilitate selecting the best compromise management alternatives.
Chapter 6 . Conclusions

This study introduced and evaluated a quantitative decision support system specifically tailored to aid forest management decisions under conflicting objectives.

Chapter 2 introduced a procedure that improves the mathematical formulation of area-based adjacency constraints (ARM) in spatially-explicit harvest scheduling models. From an Operations Research perspective, this procedure strengthens the cover inequalities of a generalized version of the Maximum Weight Stable Set problem. The results from the test runs demonstrate that the theoretically better formulations do not necessarily lead to better computational performance (i.e., shorter solution times). However, the resulting constraints may be useful for implementing in cutting plane or branch-and-cut algorithms or in combination with other types of constraints. In addition, the computational efficiency of three existing ARM formulations was compared based on their performance in solving hypothetical forest planning problems. It was found that, on average, McDill et al.’s (2002) path/cover formulation solved the test problems the quickest, followed by the proposed strengthened formulation and, finally, by Goycoolea et al.’s (2005) Maximal Clique GMU approach. As a result, McDill et al.’s (2002) original path/cover formulation was used in the multiple-objective techniques described in Chapters 3-5.
The paper in Chapter 3 compared the computational properties of four existing and one proposed method in generating the trade-offs for a bi-criteria spatially-explicit harvest scheduling problem. Computational efficiency is a very important factor in selecting a multiple-objective technique in spatially-explicit forest planning, because these are combinatorial optimization problems that are time-consuming to solve, even if only one objective is present. Based on one test run, the proposed Alpha-Delta Method displayed superior computational performance by finding the greatest number of efficient alternatives in the least amount of time.

The proposed minimum boundary patch formulation presented in Chapter 4 is an extension of two previously published works by Rebain and McDill (2003a, 2003b). This model, together with the proposed bi-objective methods in Chapter 3, allows the forest planner to indirectly but effectively control certain spatial attributes of mature forest patches that evolve as a result of harvesting activities and the growth and development over time of the management units. These spatial attributes – the shape, fragmentation and temporal connectivity – have not been modeled in prior harvest scheduling literature. Although similar approaches have been documented in the closely related field of optimal reserve selection, a key difference is that reserve selection problems are static while harvest
scheduling problems are dynamic models. In other words, once a set of reserves is selected, it is assumed that they will remain as reserves. In harvest scheduling, however, as forest stands grow, are harvested, and regenerate, patches of mature forest habitat may evolve or disappear across the landscape and over time. This additional temporal dimension requires very different modeling approaches. In addition to the spatial attributes of the evolving patches, the growth, yield and the regeneration of the forest, as well as the harvesting activities have to be incorporated into the model. The results from the test runs suggest that one of the proposed models, the TOTALMIN formulation, not only promotes the development of low perimeter-area ratio patches, but it also leads to fewer and larger patches with more temporal overlap between them. Designing harvest schedules that yield mature forest patches with advantageous spatial attributes can improve our ability to sustainably produce timber while maintaining high-quality habitat for certain wildlife species in managed landscapes.

Finally, Chapter 5 describes how four multiple-objective alternative-generating techniques, described in Chapter 3, were adapted to address discrete, three-objective forest planning problems. Moving from two- to three-dimensional discrete optimization is not as straightforward as it might seem. Although modeling the trade-offs between three or more
objectives is often necessary in integrated forest resource planning, it poses new numerical and computational challenges and therefore demands alternative mathematical programming approaches. One of the techniques discussed is entirely new and was developed by the authors, while the other, already existing methods were modified substantially in order to accommodate problems with three or more objectives. The four methods were compared, and based on this comparison it is recommended that one of the traditional techniques, the Weighted Method, should be used in combination with the proposed Alpha-Delta Method to maximize computational performance in identifying the set of efficient alternatives.

As with any research, the reader must exercise some level of caution when interpreting the computational results presented in this work. A key limitation is that only one hypothetical forest planning test problem was used in Chapter 3, 4 and 5. Although the detailed analyses of the mechanics of the discussed algorithms suggest that the observed computational results would probably apply to other problems as well, further testing is needed to confirm the relative benefits of the respective methods. Other, larger problems were formulated and solved as part of this research, but because of their larger size the Pareto-optimal set could only partially be acquired. Thus, the results of these
experimentations have not been included in this study. The results with those problems tended to support the results obtained with the problem reported here.

Computational expense is the major limitation of the work presented in this dissertation. Computational constraints limit the spatial and temporal dimensions of tractable problem size and they create numerical problems that greatly complicate the proposed multiple-objective techniques. These numerical problems can be resolved if treated properly, as discussed in Chapter 5. Although the current tractable problem size limits are smaller than the problems most practitioners want to solve, and smaller than what landscape-level forest management often demands, technological advances will continue to push these limits back. As has been pointed out in other places in this dissertation, small-scale pilot projects have an essential role in testing and fine-tuning the proposed models. They can prepare forest planners for tackling real-scale problems as the computational boundaries are pushed further back. Moreover, they can provide valuable insights into the underlying trade-offs and logical interrelationships between the competing management objectives of the problem. These insights will likely prove very useful in analyzing large-scale and computationally intractable forest planning instances.
A number of computational techniques exist that can often provide near-optimal solutions for large-scale integer programming problems even with the current computational technology and within reasonable timeframes. Some methods, such as column generation (e.g., Weintraub et al. 1994) decompose the original problem into sub-problems, then they integrate the solutions to these sub-problems in such a way that both the feasibility is preserved for the original problem, and the deviation between the optimal solution to the original problem and the aggregated optimal solutions to the sub-problems will be minimized. Another approach uses overlapping sub-problems, where the solution to one sub-problem is applied to the overlapping section of the next sub-problem and “the sub-problems are solved sequentially in a moving-windows fashion” (Hoganson and Borges 1998). Depending on the size of the windows and the amount of overlap, near-optimal solutions can be obtained for large harvest scheduling problems with relatively low computational effort. Although Hoganson and Borges (1998) formulated and solved the sub-problems using dynamic programming, the same ‘moving-windows’ concept could be applied to the multi-objective 0-1 programs discussed in this dissertation. A third, hybrid approach is being developed by McDill et al. (unpublished) that uses non-spatial linear-programming models to model long-term, large-scale (district-wide) concerns to set harvest
area targets, and then uses spatially-explicit integer programming models to identify specific harvest units to cut in each planning period to meet the targets determined by the LP models.

As with all heuristic approaches, none of the above methods can guarantee optimal solutions, and in most cases only crude estimates of the deviation from optimality are available. Again, this can lead to serious numerical problems for multiple-objective problems (see Chapter 5 for details). Thus, solving large-scale (those with more than a few hundred units), multiple-objective harvest scheduling problems is only possible by forgoing Pareto-optimality for some or most of the alternatives.

Computational boundaries also limit our ability to integrate forest management objectives that require model formulations at different spatial or temporal scales. Addressing this issue from the mathematical modeling perspective is important because the need to integrate management plans across various scales occurs frequently in practice. While profit maximization, timber production or recreation are typically district-wide concerns (for example, the state forests of Pennsylvania comprise about 500-2,500 management units), wildlife habitat objectives often need to be defined at smaller or larger scales. The following example illustrates one mathematical modeling option for these
types of forest planning situations. Suppose there are only two objectives, one is profit maximization, which must be addressed at the district level (1,500 units), and the other is mature forest habitat maximization, which is defined at the landscape level (about 250 units). The model formulation would also have two objectives functions: (1) to maximize profit for the entire district, and (2) to maximize the minimum amount of mature forest habitat over the six landscapes that comprise the district. If computational boundaries do not allow this approach, then separate bi-objective programs would need to be formulated and solved for each of the six landscapes. Each program would have two objectives: profit and habitat maximization – both defined at the landscape level. After solving the bi-objective programs, column generation (Weintraub et al. 1994) or the “moving-windows” technique (Hoganson and Borges 1998), discussed in the previous paragraph, could be used to integrate the solutions to these “sub-problems” at the larger, district-wide level.

Another limitation of the proposed multiple-objective methods stems from the fact that not all forest benefits are quantifiable. The aesthetic or spiritual values of forests are hard to measure using exact mathematical terms. Still, they are very important to many people. The trade-offs between these quantifiable and non-quantifiable objectives cannot be quantified either. Although the decision support system proposed in this work can only
identify the trade-offs between quantifiable objectives, it can still serve as a framework to address objectives that are hard to define in exact terms. The set of Pareto-optimal management alternatives could be generated with respect to the quantifiable objectives first, then, if the stakeholders cannot make a decision, then an additional, possibly unquantifiable, criterion may be used to tip the balance. There is no guarantee, however, that the resulting solution will be Pareto-optimal with respect to this additional criterion.

Clearly, the proposed decision support methods should be used in concert with other techniques such as methods to visualize the aesthetic impacts of various harvest schedules and consensus-building tools like the Delphi Process or the Nominal Group Technique (Delbecq et al., 1975). Along with computational constraints, visualizing the trade-offs and the efficient management alternatives becomes an issue once the number of management objectives exceeds three. Again, small-scale pilot studies could determine what approach would work best with the public or private constituents. A clear, unbiased presentation of management alternatives and their impact on the landscape is critical to successful forest planning. Thus, visualization must be a high-priority issue in future research.

The decision support tools introduced in this work need to be applied and fine-tuned to on-the-ground forest management planning problems with various public and private
constituents. Learning how stakeholders and private landowners use the information that is provided about the trade-offs and production possibilities of a problem might help forest resource professionals better manage the conflicts that inevitably arise in many private and public forest management decisions.
References


Tóth, S.F., M.E. McDill and S. Rebain. Forthcoming. Finding the Efficient Frontier of a Bi-

solving general forest planning problems with adjacency constraints. For. Sci. 40:142-161.


Williams, J. C. 1998. Delineating protected wildlife corridors with multi-objective
programming. Envir. Mod. and Assess. 3:77-86.

Williams, J.C. and C.S. ReVelle. 1996. A 0-1 programming approach to delineating
protected reserves. Envir. Planning B: Planning and Design. 23:607-624


Vita

Sándor Ferenc Tóth


With a Japanese Government (Monbusho) Research Scholarship, Mr. Tóth spent three years pursuing graduate and language studies at the Shinshu and Hitotsubashi universities (Japan). He was awarded a Master of Agriculture degree in March 2002. The author enrolled in a Dual Title Doctor of Philosophy Program in Forest Resources and Operations Research at the Pennsylvania State University in May 2002. He completed the degree in August 2005.

Mr. Tóth is member of the Society of American Foresters, the Japan Society of Forest Planning, the Hungarian Forestry Association, Xi Sigma Pi, the Institute for Operations Research and the Management Sciences, and the International Society on Multiple Criteria Decision Making. He is currently employed as a Research Associate at the Pennsylvania State University.

He is married to the former Amy Marie Nearhoof of Warriors Mark, Pennsylvania.