AN ABSTRACT OF THE THESIS OF

Atsushi Yoshimoto for the degree of Doctor of Philosophy in Forest Resources presented on June 1, 1990.

Title: Economic Analysis Of Integrating Spatial Concerns Into Harvest Scheduling.

Abstract approved: Signature redacted for privacy.
J. Douglas Brodie

An analytical algorithm to formulate a sparser set of adjacency constraints than the conventional algorithm is proposed. Utilizing matrix algebra applied to an adjacency matrix, constraints are systematically and efficiently derived by the proposed algorithm. The proposed algorithm is proved to provide a true set of adjacency constraints in the sense that no adjacent harvests can occur.

A heuristic technique to solve a spatially constrained area-based harvest scheduling problem with even-flow constraints is proposed. The technique combines the modified random search technique, the modified binary search method, and the PATH algorithm. Partitioning the multiperiod scheduling problem, period by period using the PATH algorithm, the objective at each period is respecified by minimizing harvest flow fluctuation from the lower bound of the harvest flow level, and the feasibility of a solution at each period is expanded to both the current and the

following periods. The modified random search technique is applied to generate a feasible solution at each period. The modified binary search method is used to obtain an optimal or appropriate even-flow level.

By using the proposed heuristic technique, the cost evaluation of implementing various spatial restrictions on riparian zone planning is presented.

Economic Analysis Of Integrating Spatial Concerns Into Harvest Scheduling

by

Atsushi Yoshimoto

A THESIS

submitted to

Oregon State University

in partial fulfillment of

the requirement of

the degree of

Doctor of Philosophy

Completed June 1, 1990

Commencement June 1991

To

my wife, Yoko

and

my parents, Manabu and Setsuko

ACKNOWLEDGEMENTS

The author would like to express his gratitude to those who gave assistance on this dissertation. He especially acknowledges his major professor, Dr. J. Douglas Brodie, for advice and patient review and editing of all manuscripts in this dissertation. He would also like to express his appreciation to Dr. J. Sessions, Dr. A. Love, and Dr. S. Daniels for serving on his committee, and Dr. T. Schowalter for representing the Graduate School. He thanks Juan M. Torres R. for sharing ideas and discussing the problems. Final appreciation is expressed to his wife, Yoko, for her helping him concentrate on this work and study comfortably and efficiently.

TABLE OF CONTENTS

	Page
Chapter 1: INTRODUCTION	1
Chapter 2: COMPARATIVE EFFICIENCY OF ALGORITHMS TO GENERATE ADJACENCY CONSTRAINTS	5
ABSTRACT	5
INTRODUCTION	7
REVIEW OF ALGORITHMS	9
AN ANALYTICAL ALGORITHM	20
COMPARISON OF EFFICIENCY OF ALGORITHMS	30
CONCLUSIONS	44
LITERATURE CITED	47
Chapter 3: A COMPARISON OF APPROACHES FOR INTEGRATING SPATIAL CONCERNS INTO HARVEST SCHEDULING	50
ABSTRACT	50
INTRODUCTION	52
THE SPATIALLY CONSTRAINED HARVEST SCHEDULING PROBLEM	57
Integer Programming Formulation	57
The Random Search Technique	61
The Composite Relaxation Technique	65
THE ALTERNATIVE TECHNIQUE BASED ON A RANDOM SEARCH TECHNIQUE	70
Projection Alternative TecHnique (PATH) in Harvest Scheduling Problems	70
Minimizing Deviation of Harvest Flow	73
The Binary Search Method For The Harvest Flow Level	77
Maximizing Present Net Worth	78
MODEL DEVELOPMENT	85
MODEL EXPERIMENTATION	88
EVALUATION OF THE PERFORMANCE OF THE PROPOSED ALGORITHM	108
Solutions By The Branch-and-Bound Algorithm	108
Solutions By The Composite Relaxation Technique	115
Comparison Of Different Techniques	118
CONCLUSIONS	125
LITERATURE CITED	128
Chapter 4: COST EVALUATION OF IMPLEMENTING SPATIAL RESTRICTIONS ON RIPARIAN ZONE PLANNING IN WESTERN OREGON	133
ABSTRACT	133
INTRODUCTION	135
THE SCHEDULING SYSTEM OF MANAGEMENT ALTERNATIVES FOR TIMBER-HARVEST (SSMART)	139
SITE SELECTION AND DATA DESCRIPTION	144

TABLE OF CONTENTS (continued)

COST EVALUATION OF SPATIAL RESTRICTIONS 149
Static Change In Buffer Size 152
Dynamic Change In Buffer Relationship 161
Change In Adjacency Lag Periods 165
Evaluating Marginal Cost 167
CONCLUSIONS 172
LITERATURE CITED 175

Chapter 5: CONCLUSIONS 178

BIBLIOGRAPHY 182

LIST OF FIGURES

Figure		Page
2.1.	A five unit numerical example	11
2.2.	Three patterns for a type 1 inequality a) Pattern 1 (pair) b) Pattern 2 (triplet) c) Pattern 3 (quadruplet)	11
2.3.	Two example structures a) 23 unit problem b) 20 unit problem	31
3.1.	The structure of ROHO	76
3.2.	The structure of the ROHO-PATH algorithm	83
3.3.	The first example forest from Nelson et al. (1990)	89
3.4.	The stand age distribution a) The first example forest (Nelson et al. 1990) b) The second example forest (Barker 1989)	90
3.5.	The second example forest from Barker (1989)	92
3.6.	The relation between the objective value and even-flow level V0 for the first example forest by SSMART a) NI = 1 b) NI = 3	95
3.7.	Changing patterns of flow fluctuation for the first example forest by SSMART a) NI = 1 b) NI = 3	96
3.8.	Comparison of solutions by SSMART for the first example forest a) In terms of the objective value b) In terms of the computational time	98
3.9.	The relation between the objective value and even-flow level V0 for the second example forest by SSMART a) NI = 1 b) NI = 3	100

LIST OF FIGURES (continued)

3.10. Changing patterns of flow fluctuation for the second example forest by SSMART
a) NI = 1
b) NI = 3 102

3.11. Comparison of solutions by SSMART for the second example forest
a) In terms of the objective value
b) In terms of the computational time 103

3.12. Dynamics of stand age distribution of the first example forest over 10 periods by SSMART 105

3.13. Dynamics of stand age distribution of the second example forest over 10 periods by SSMART 106

3.14. Comparison of techniques for the first example forest
a) In terms of the objective value
b) In terms of the computational time
c) In terms of the flow fluctuation 119

3.15. Comparison of techniques for the second example forest
a) In terms of the objective value
b) In terms of the computational time
c) In terms of the flow fluctuation 122

4.1. An example forest with 142 stands 145

4.2. Stand age distribution for an example forest 146

4.3. The relationship between an aquatic area and the riparian management area 150

4.4. Results from the static-scenario by the approach without the use of SSMART to constrained problems
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost 153

4.5. Results from the static-scenario by the approach with the use of SSMART to constrained problems
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost 156

LIST OF FIGURES (continued)

4.6. Results from the dynamic-scenario
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost 163

4.7. Results from the lag-scenario
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost 166

LIST OF TABLES

Table		Page
2.1.	Type 1 and possible Type 2 patterns for a five unit problem	15
2.2.	Selection of final constraints for a five unit problem	15
2.3.	Adjacency constraints derived from each algorithm	29
2.4.	Adjacency constraints for the 23 unit problem	33
2.5.	Adjacency constraints for the 20 unit problem	34
2.6.	A comparison of the efficiency of the algorithms	35
2.7.	Return from each unit for both 23 and 20 unit problems	40
2.8.	Results of computational comparisons among algorithms	41
3.1.	Input data file for SPS (see Arney 1985)	91
3.2.	The final solutions for the first example forest by SSMART	99
3.3.	The final solutions for the second example forest by SSMART	104
3.4.	The final solutions for the first example forest by LINGO/386	111
3.5.	The final solutions for the first example forest by RELAX	117
3.6.	Comparison of solutions by SSMART and LINGO/386	124
4.1.	Initial stand condition for SPS (see Arney 1985)	147
4.2.	Summary of results from the static-scenario in the first approach - No use of SSMART to constrained problems	155
4.3.	Summary of results from the static-scenario in the second approach - SSMART applied to all constrained problems	158

LIST OF TABLES (continued)

4.4. Comparison of the actual flow fluctuation in the static-scenario 160

4.5. Summary of results from the dynamic-scenario 164

4.6. Summary of results from the lag-scenario 168

4.7. Marginal cost for different scenarios 170

ECONOMIC ANALYSIS OF INTEGRATING SPATIAL CONCERNS INTO HARVEST SCHEDULING

Chapter 1

INTRODUCTION

In recent years, traditional and newer demands for use of forest resources have increased and allocation decisions have become controversial. The demands and expectations for utilization of forests have accelerated and conflicts focus needs for efficient allocation. Actions to protect wildlife habitat and/or a stream side, vegetation and soils, for example, have restricted harvesting, resulting in a need for careful decision processes for managing forest resources.

Due to the complexity of the problems, mathematical programming techniques have played an important and useful role in seeking solutions. The techniques simultaneously allocate activities, such as road construction, protection, and silvicultural treatment, within the harvest scheduling decision framework. Linear programming (LP) solution techniques have been widely used in the decision making process in harvest scheduling models. Most LP forest planning applications require extensive aggregation to meet dimensionality restrictions, and decision variables therefore require disaggregation before they can be implemented. Because of aggregation the optimal solution from LP can be infeasible, inferior, or both, when compared

with disaggregated approaches.

Integer optimization deals with problems of maximizing or minimizing a function of many decision variables subject to inequality and equality constraints and integer restrictions on some or all of the decision variables. A rich variety of problems, therefore, can be represented by these discrete optimization models.

In order to obtain more precise solutions in harvest scheduling problems, particularly those with spatial concerns, many decision variables are required to be 0-1 integer. Forest managers can implement these solutions knowing that they are spatially feasible. Because of this requirement on the ground, the 0-1 integer programming solution techniques have been replacing LP based models, for problems with transportation and spatial concerns.

Multiple-use concerns and environmental requirements can be taken into account by imbedding numerous constraints in the problem formulation. Adjacency constraints are of current concern in harvest scheduling. These constraints require no adjacent harvest units, which permits limitation of clearcut block size and leads to a discrete optimization problem. The first manuscript, Chapter 2, describes the development of an efficient algorithm to formulate adjacency constraints, which can be used in integer programming solution techniques or in heuristic techniques. This algorithm utilizes matrix algebra to formulate nonlinear

adjacency constraints from an adjacency matrix. The algorithm converts such nonlinear constraints into linear constraints by taking advantage of 0-1 restrictions on the decision variables.

Imbedding adjacency constraints as well as even-flow constraints, harvest scheduling problems become difficult to solve by the exact solution techniques, such as the branch-and-bound algorithm. The second manuscript, Chapter 3, addresses the problem of solving a spatially constrained area-based harvest scheduling problem with even-flow constraints. A heuristic technique is developed to provide a "good" integer solution. The technique combines the modified random search technique, the modified binary search method, and the PATH algorithm. Since the even-flow constraints are most likely to be violated by the integer solution, the proposed technique respecifies the problem by minimizing a harvest flow fluctuation as well as maximizing the present net worth of the sum of returns from harvest units. A harvest scheduling model called SSMART (Scheduling System of Management Alternatives foR Timber-harvest) is developed. A comparison of this technique with several different techniques is provided.

The third manuscript, Chapter 4, addresses the problem of evaluating the cost of implementing spatial restrictions on riparian zone planning. Three scenarios are addressed. The first scenario prohibits harvest in the riparian

management area, while the second scenario allows harvest with additional adjacency restrictions among segments of the riparian area. Instead of prohibiting harvest in the riparian management area, the third scenario utilizes additional adjacency lag periods as alternative spatial restrictions to harvest scheduling. Because of its capability to efficiently solve the long-term scheduling problem, SSMART is used to implement the cost evaluation.

Chapter 2

COMPARATIVE EFFICIENCY OF ALGORITHMS TO GENERATE ADJACENCY CONSTRAINTS

by

Atsushi Yoshimoto

and

J. Douglas Brodie

ABSTRACT

A mathematical programming formulation of the area-based forest planning problem can result in a large number of adjacency constraints with much potential for redundancy. Two heuristic algorithms have been proposed for reducing redundant adjacency constraints generated by the conventional algorithm. In this paper another analytical algorithm is proposed, and its efficiency and that of the conventional algorithm and the two heuristics are evaluated and compared. Comparison is based on the number of constraints, and on the computational effort needed both to derive the adjacency constraints and to solve the associated planning problem. Evaluation for several adjacency maps shows that the conventional algorithm has the largest number of constraints with a low degree of effort in derivation of

adjacency constraints and a small computational task to find a final solution. The first heuristic algorithm has the smallest number of constraints but involves a high degree of effort and a large computational task. The second heuristic has a small number of constraints with a moderate degree of effort and a large computational task, and the proposed algorithm has a small number of constraints with a low degree of effort and a moderate to large computational task.

INTRODUCTION

Harvest scheduling problems are usually formulated using mathematical programming techniques. Because of multiple-use concerns, environmental requirements are taken into account by imbedding numerous constraints in the formulation. One of the current concerns is adjacency constraints, which require no clearcutting among adjacent areas during the same harvesting period. These adjacency constraints disperse clearcut areas throughout the forest, providing a landscape mosaic and limiting the size of individual harvest blocks. Thompson et al. (1973) utilized adjacency constraints to distribute clearcuts in a linear programming formulation. Instead of using spatial consideration as explicit constraints in the formulation, Mealey et al. (1982) introduced decision variables containing a scheduling package, which met spatial considerations, in the linear programming formulation. Hokans (1983) incorporated adjacency constraints into an artificial intelligence procedure. Recently efforts have been made to incorporate spatial concerns into applied mathematical programming or heuristic algorithms for solving applied harvest scheduling problems (Gross and Dykstra 1988, Gross 1989, Nelson et al. 1988, Sessions and Sessions 1988).

From the viewpoint of the mathematical programming formulation, adjacency concerns require a large number of

constraints using the conventional algorithm. The conventional algorithm consists of writing a pairwise adjacency constraint for every adjacency relationship. Due to limitations on the number of constraints in linear programming commercial software, the reduction of the number of constraints plays a critical role in formulating even modest sized problems. Two heuristic algorithms which generate a sparser set of adjacency constraints were introduced by Meneghin et al. (1988) and Torres and Brodie (1990). Although the number of constraints is reduced dramatically, the procedures and rules of both heuristics are difficult to follow and understand.

The purpose of this paper is to propose a simple analytical algorithm to generate adjacency constraints which can be proved to ensure that no adjacent harvest occurs, and that is easier to understand than the heuristics. Then we will compare the efficiency of each algorithm in terms of the number of constraints, the degree of effort needed to create the final constraints, and the computational burden encountered in solving the associated planning problem. In the first section, the conventional algorithm, the algorithm by Meneghin, Kirby and Jones (1988) (M-K-J algorithm), and the algorithm by Torres and Brodie (1990) (T-B algorithm) are reviewed. In the second section, the analytical algorithm is derived. Then in the third section, a comparison of the four algorithms is presented.

REVIEW OF ALGORITHMS

Our problem is concerned with the selection of harvest units to meet an objective with adjacency constraints. The objective can be to maximize present net worth of return from harvest units, to maximize harvest volume, etc. Since the decision is to select a unit or not to select a unit (harvest or not harvest), the control variable, X_i for the i-th unit, is dichotomous or zero-one defined by:

$$X_i = \begin{cases} 1 \text{ if the i-th unit is selected} \\ 0 \text{ otherwise} \end{cases} \qquad (1)$$

After introducing dichotomous control variables, adjacency constraints can be formulated in several ways. In what follows, three methods: the conventional algorithm, the M-K-J algorithm, and the T-B algorithm are reviewed.

The conventional algorithm is widely used in mixed integer programming or integer programming formulations. Each constraint takes into account only two units adjacent to each other. The number of constraints is therefore determined by the number of combinations of these two adjacent units. The pairwise formulation is applied:

$$X_i + X_j \leq 1 \quad i < j \qquad (2)$$

where the i-th unit and the j-th unit are adjacent. The inequality (2) implies that one of the following decisions is made:

1. the i-th unit is selected, not the j-th unit,

2. the j-th unit is selected, not the i-th unit,

or 3. neither is selected.

Applied to the simple spatial example shown in Figure 2.1, the conventional algorithm provides the following adjacency constraints:

$$\begin{aligned} X_1 + X_2 &\le 1 \\ X_1 + X_4 &\le 1 \\ X_2 + X_3 &\le 1 \\ X_2 + X_4 &\le 1 \\ X_3 + X_4 &\le 1 \\ X_3 + X_5 &\le 1 \\ X_4 + X_5 &\le 1 \end{aligned} \qquad (3)$$

The advantage of this algorithm is that it is easy to formulate and it constructs a "true" set of constraints, in a sense that no adjacent harvests occur. The disadvantage is that many redundant constraints are formulated.

The M-K-J algorithm was introduced to reduce the number of adjacency constraints by using the three patterns (pair, triplet, quadruplet) of adjacency structure depicted in Figure 2.2. The M-K-J algorithm works as follows. At first, a set of inequalities that were named type 1 inequalities is set up. A type 1 inequality is formulated by one of the following forms:

Pair : $X_i+X_j \le 1$ (pattern 1 in Figure 2.2) (4)

Triplet : $X_i+X_j+X_k \le 1$ (pattern 2 in Figure 2.2) (5)

Quadruplet: $X_i+X_j+X_k+X_l \le 1$ (pattern 3 in Figure 2.2) (6)

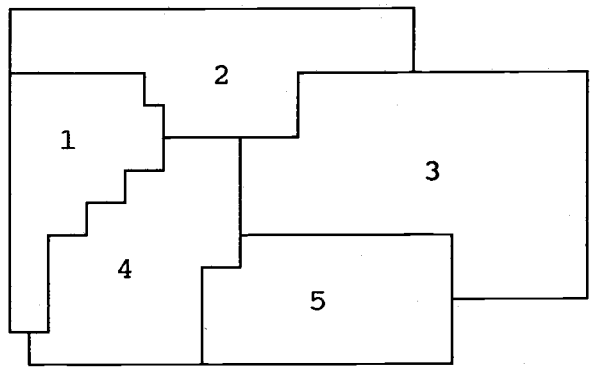


Figure 2.1. A five unit numerical example

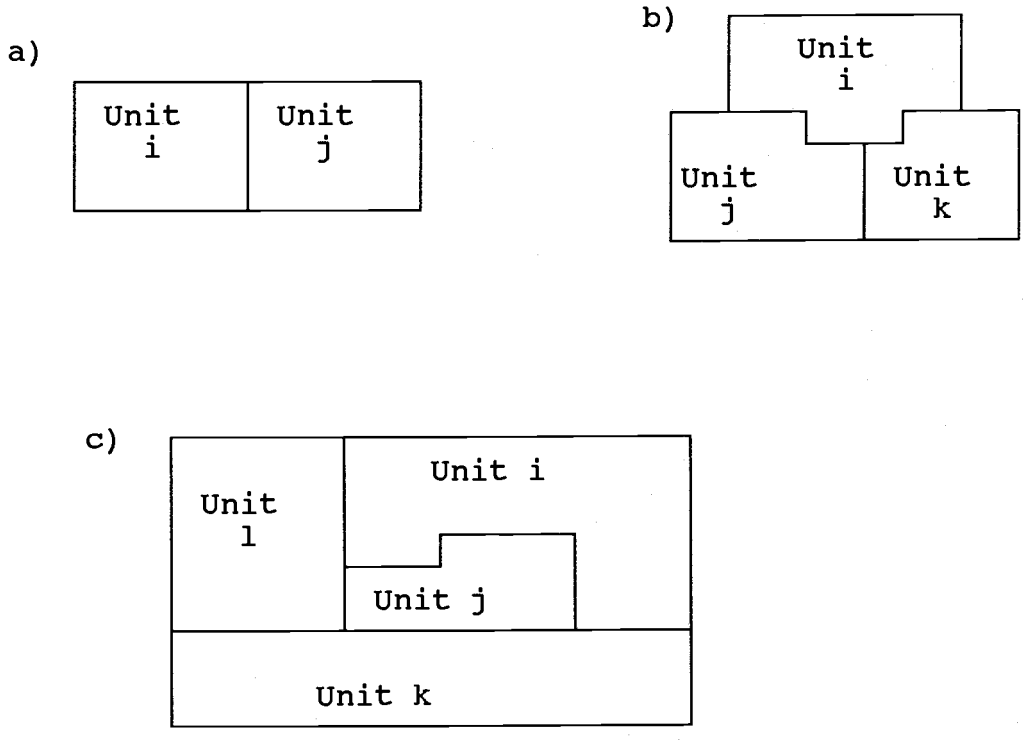


Figure 2.2. Three patterns for a type 1 inequality
a) Pattern 1 (pair)
b) Pattern 2 (triplet)
c) Pattern 3 (quadruplet)

After type 1 inequalities are specified, combining a certain set, say **T1**, of type 1 inequalities creates an aggregated inequality called a type 2 inequality.

Aggregation of type 1 inequalities in **T1** follows the following rule. Each type 1 inequality in **T1** can be formulated by either

$$\sum_{i\epsilon E} X_i + X_j \leq 1, \quad j \epsilon P \tag{7}$$

or

$$\sum_{i\epsilon E} X_i + \sum_{j\epsilon G} X_j \leq 1 \tag{8}$$

where index sets E, P and G are defined by:

1. if all inequalities in **T1** have X_i, index i belongs to a set E

2. if there exists only one element X_j in one type 1 inequality in **T1** beside X_i ($i\epsilon E$), index j belongs to a set P

3. if there exist more than one X_j element in one type 1 inequality in **T1** beside X_i ($i\epsilon E$), the j indices belong to a set G.

Then using r as the total number of type 1 inequalities in **T1**, one type 2 inequality can be formulated as:

$$(2r-1) \sum_{i\epsilon E} X_i + \sum_{i\epsilon P} X_i + r \sum_{i\epsilon G} X_i \leq (2r-1) \tag{9}$$

Notice that at most one type 1 inequality in **T1** can be formulated by inequality (8). Since a coefficient (2r-1) of X_i ($i\epsilon E$) is the same as (2r-1) at the right-hand side (RHS), when one of the X_i's ($i\epsilon E$) becomes 1, all other units are forced to be zero. A coefficient r of X_i ($i\epsilon G$) doesn't allow more than one unit of X_i's ($i\epsilon G$) to be 1 at the same time because $r \leq 2r-1 < 2r$. However, it does allow one unit

of X_i's ($i \epsilon G$) and any combination of units in a set P to be 1 at the same time because the number of units in a set P is less than or equal to (r-1). Thus, one type 2 inequality provides all possible selections of units in **T1**. After creating all possible combinations of type 1 inequalities to generate one type 2 inequality, the final set of type 2 inequalities are specified in accordance with the heuristic procedure provided by Meneghin et al. (1988). Their heuristic procedure requires that all possible combinations of type 1 inequalities be set up for one type 2 inequality. Among them the final constraints are determined following four rules. The rules are made in order to combine all type 1 inequalities into type 2 inequalities. Since a set of type 1 inequalities are generated to cover all adjacency relationships, so does a corresponding set of type 2 inequalities. It can be proved, therefore, that as long as rules are set up to include all type 1 inequalities, the final set of constraints, type 2 inequalities, constructs a true set of adjacency constraints.

Using the example in Figure 2.1, the procedure of the M-K-J algorithm is described as follows. Type 1 inequalities are to be identified first. Following their procedures, a lower triangular of the adjacency matrix is first derived as follows:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

The adjacency matrix is an (n x n) square matrix whose element a_{ij} is defined by:

$$a_{ij} = \begin{cases} 1 & \text{if the i-th unit is adjacent to the j-th unit} \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

and n is the number of units. The adjacency matrix represents all adjacency relationships among units, and so does a lower triangular of the adjacency matrix. In the M-K-J algorithm, first, select a_{21} and look for other units adjacent to both unit 1 and unit 2. Unit 4 is adjacent to them since $a_{41} = a_{42} = 1$. Thus unit 1, 2, and 4 construct a triplet:

$$\text{I: } X_1 + X_2 + X_4 \leq 1 \quad (11)$$

Second, select a_{32}. Since $a_{42} = a_{43} = 1$, then unit 4 is also adjacent to both unit 2 and 3, resulting in another triplet:

$$\text{II: } X_2 + X_3 + X_4 \leq 1 \quad (12)$$

Third, select a_{43}. Since $a_{53} = a_{54} = 1$, unit 5 is adjacent to both unit 3 and 4. Then the last type 1 inequality is:

$$\text{III: } X_3 + X_4 + X_5 \leq 1 \quad (13)$$

After specifying type 1 inequalities, all possible combinations of type 1 inequalities that construct a type 2 inequality are specified (see Table 2.1).

Table 2.1. Type 1 and possible Type 2 patterns for a five unit problem

Type 1 label	Member	Possible Type 2	Type 2 Frequency
I	1-2-4	(I,II)	1
II	2-3-4	(I,II),(II,III)	2
III	3-4-5	(II,III)	1

For example, combining I and II type 1 inequalities, r is set as 2, E = {2,4}, P = {1,3} and there are no indices in G, resulting in a type 2 inequality:

$$3X_2 + 3X_4 + X_1 + X_3 \leq 3. \qquad (14)$$

Ascending order based on type 2 frequency reallocates type 1 patterns as in Table 2.2.

Table 2.2. Selection of final constraints for a five unit problem

Type 1 label	Possible Type 2	Choice	Final Constraints
I	(I,II)	(I,II)	3X2 + 3X4 + X1 + X3 ≤ 3
III	(II,III)	III	X3 + X4 + X5 ≤ 1
II	(I,II),(II,III)		

Choice is made in accordance with procedures by Meneghin et al. (1988). The final type 2 inequalities are:

$$3X_2 + 3X_4 + X_1 + X_3 \leq 3. \qquad (14)$$

and

$$X_3 + X_4 + X_5 \leq 1. \qquad (15)$$

For this example, the number of type 1 inequalities is 3 and the number of all possible combinations of type 1 inequalities for type 2 inequalities is 4.

The advantage of the M-K-J algorithm is that once a set of type 1 inequalities, **T1**, is specified for one type 2 inequality, it can be proven that the type 2 inequality represents all type 1 inequalities in **T1**, resulting in a reduction of the number of constraints. The difficulty is that it takes too much time to identify all type 1 inequalities, combine them to make all possible type 2 inequalities for the final constraints, and then specify the final constraints.

The T-B algorithm utilizes the technique of penalization and the four-color conjecture. The T-B algorithm first eliminates the obvious redundant constraints by using an adjacency matrix. This is called Procedure 1. Procedure 1 works as follows. Using n as the total number of units and a_{ij} as an element of an adjacency matrix at the i-th row of the j-th column, if

$$\sum_{j=1}^{n} a_{ji} = \sum_{j=1}^{n} a_{ij}, \quad i = 1,,,n \tag{16}$$

we know that the i-th row and the i-th column represent the same adjacency relationships. In other words, if $a_{ji} = 1$, then $a_{ij} = 1$. Thus we can replace a_{ij} by 0 at the i-th row in the matrix without missing any adjacency relationships[1]. After eliminating redundant rows, one adjacency constraint

---

[1]Torres and Brodie (1990), however, did not prove this. Instead they warned that if Procedure 1 is applied, there is a possibility of violating some adjacency relationships. The proof is presented in the following section.

is set up for each row not eliminated in Procedure 1 (the unit corresponding to this row is called a reference unit). This is referred to as Procedure 2. Torres and Brodie (1990) provide heuristic rules to create an adjacency constraint for each reference unit. The rules rely on the technique of penalization. Unlike the pair, triplet, and quadruplet relationships in Figure 2.2, a general adjacency relationship of the reference unit to surrounding units requires penalization on each coefficient of the control variables in an adjacency constraint. The concept of the "Four Color Conjecture" (May 1965) and a suggested "coloring number" of five was introduced to support their use of penalization. However, no theoretical derivation of use of a coloring number of five instead of four was provided except the statement "...by using larger coloring numbers we reduce that possibility [possibility of violating adjacency relations for complicated patterns].." This statement casts some doubt on the applicability of their heuristic for many types of patterns. Although Procedure 1 to eliminate redundant rows can be proved correct and sufficient to generate adjacency constraints, Procedure 2 remains heuristic. The following describes the procedure for penalization of a coefficient for a control variable in an inequality from the example in Figure 2.1.

Following Torres and Brodie (1990), Procedure 1 is applied to eliminate redundant rows in an adjacency matrix

for the example in Figure 2.1, resulting in:

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

The first and third rows are redundant. Following Procedure 2, units 2, 4, and 5 are selected as the reference unit. After bringing adjacent units to a reference unit into a constraint then penalizing coefficients of the control variables, the following three constraints are obtained. The adjacency constraint for the reference unit 2 is:

$$6X_2 + 3X_1 + 4X_4 + 2X_3 \leq 6, \tag{22}$$

for the reference unit 4:

$$7X_4 + 3X_1 + 5X_2 + 4X_3 + 2X_5 \leq 7, \tag{23}$$

for the reference unit 5:

$$3X_5 + 2X_4 + 2X_3 \leq 3. \tag{24}$$

In order to obtain the final constraints the T-B algorithm requires 9 entering units and 18 identical inequalities of which equations (22), (23) and (24) require 6, 9 and 3 inequalities, respectively.

As can be seen in Meneghin et al. (1988) and Torres and Brodie (1990), both algorithms utilize heuristic procedures or rules for creating a smaller set of adjacency constraints than results from application of the conventional algorithm. However, it takes more time to set up adjacency constraints by following their procedures than to use the conventional

formulation. In what follows, an analytical algorithm is proposed. The algorithm creates a sparser set of constraints than the conventional algorithm and requires less time to implement than either the M-K-J or T-B algorithms.

AN ANALYTICAL ALGORITHM

In this section, we derive a new algorithm to write adjacency constraints by using matrix algebra. Then we apply the proposed algorithm to the example in Figure 2.1 in order to demonstrate how it works.

As defined in the previous section, let **A** be an (n x n) adjacency matrix, X_i be a dichotomous control variable for the i-th unit, and n be the number of units. In addition, introduce an (n x 1) control vector $\mathbf{X} = (X_1, X_2, \cdots, X_n)^t$ where t represents the transpose. Further introduce a new vector called an adjacency vector **V** which is defined by the product of an adjacency matrix **A** and a control vector **X**:

$$\mathbf{V} = \mathbf{A} \cdot \mathbf{X} \tag{25}$$

The i-th element of the adjacency vector, v_i, is the sum of X_j's where the j-th unit is adjacent to the i-th unit:

$$v_i = \sum_{j \in \mathbf{S}_i} X_j \tag{26}$$

$\mathbf{S}_i$ is an index set called an adjacent set for the i-th unit defined by:

$\mathbf{S}_i$ = { all j:such that the j-th unit is adjacent to the i-th unit}

Adjacency constraints ensure that if two units are adjacent, at least one of them can not be selected for harvest. Thus the product of their 0-1 control variables is always zero in any feasible solution. In other words, $X_i \cdot X_j = 0$ when the i-th unit and the j-th unit are adjacent. This

is true also for all $j \epsilon \mathbf{S}_i$. Thus we have the following equation:

$$\sum_{j \epsilon \mathbf{S}_i} X_i \cdot X_j = 0$$

or

$$X_i \cdot [\sum_{j \epsilon \mathbf{S}_i} X_j] = 0 \qquad (27)$$

Equation (27) implies the following:

1. if $X_i = 1$, all X_j's ($j \epsilon \mathbf{S}_i$) are zero,
2. if any $X_j = 1$ ($j \epsilon \mathbf{S}_i$), $X_i = 0$,
3. X_i and X_j's ($j \epsilon \mathbf{S}_i$) can all be zero.

Therefore equation (27) represents all adjacency relationships of the i-th unit to its surrounding units. This is easily extended leading to the adjacency constraints for all units by a matrix equation as follows:

$$\mathbf{X}^t \cdot \mathbf{V} = \underline{\mathbf{0}} \qquad (28)$$

where $\underline{\mathbf{0}}$ is an (n x 1) zero vector, and a control vector $\mathbf{X}$ and an adjacency vector $\mathbf{V}$ are orthogonal. In order to convert nonlinear adjacency constraints from equation (28) to linear adjacency constraints, we go back to equation (27) again.

Introducing r_i as the number of indices in $\mathbf{S}_i$, we can convert equation (27) into the following:

$$\sum_{j \epsilon \mathbf{S}_i} X_j \leq r_i \cdot [1 - X_i]$$

or

$$r_i \cdot X_i + \sum_{j \epsilon \mathbf{S}_i} X_j \leq r_i \qquad (29)$$

Actually, r_i can be calculated by:

$$r_i = \mathbf{A}_{i\bullet} \cdot \mathbf{1} \qquad (30)$$

where $\mathbf{A}_{i\bullet}$ is the i-th row vector of an adjacency matrix **A** and **1** is an (n x 1) unit vector. Equation (29) also implies the same cases as equation (27) does.

For the purpose of mathematical formulation, introduce an (n x n) diagonal matrix **B** in which the i-th diagonal element b_{ii} is defined by:

$$b_{ii} = r_i = \mathbf{A}_{i\bullet}\cdot\mathbf{1} \tag{31}$$

Thus because $\mathbf{A}_{i\bullet}\cdot\mathbf{X} = \sum_{j\in\mathbf{S}_i} X_j$, equation (29) can be expressed by:

$$b_{ii}\cdot X_i + \mathbf{A}_{i\bullet}\cdot\mathbf{X} \leq \mathbf{A}_{i\bullet}\cdot\mathbf{1} \tag{32}$$

For all units i's, we have the true adjacency constraints expressed by:

$$\mathbf{B}\cdot\mathbf{X} + \mathbf{A}\cdot\mathbf{X} \leq \mathbf{A}\cdot\mathbf{1}$$

or

$$\mathbf{M}\cdot\mathbf{X} \leq \mathbf{A}\cdot\mathbf{1} \tag{33}$$

where $\mathbf{M} = [\mathbf{A} + \mathbf{B}]$ called a modified adjacency matrix defined by $\{m_{ij}\}$ where:

$$m_{ij} = \begin{cases} a_{ij} & \text{if } i \neq j \\ \mathbf{A}_{i\bullet}\cdot\mathbf{1} & \text{if } i = j \end{cases} \tag{34}$$

As a result, we can derive the following proposition.

Proposition: If **A** is an (n x n) adjacency matrix with an (n x n) modified adjacency matrix **M**,

$$\mathbf{M}\cdot\mathbf{X} \leq \mathbf{A}\cdot\mathbf{1} \tag{33}$$

represents all adjacency constraints.

Based on equation (33), it is possible to further reduce both the number of control variables in each constraint and the number of constraints, while retaining all adjacency relationships in the matrix. To reduce the

number of control variables in each constraint, the nature of symmetry of an adjacency matrix **A** is utilized. Since **A** is symmetric, i.e., $\mathbf{A}^t = \mathbf{A}$, an upper triangular and a lower triangular of the matrix of **A** represent the same adjacency relationships as **A** itself does. This is due to the fact that $a_{ij} = a_{ji}$, implying that $X_i \cdot X_j = X_j \cdot X_i = 0$. Eliminating elements above or below the principal diagonal of **A**, an upper or lower triangular matrix of **A** is obtained (call this as a triangular adjacency matrix (TAM)). This TAM can also be applied to the above proposition with the corresponding modified matrix to generate adjacency constraints.

For further reduction of the number of constraints from **A**, Procedure 1 in Torres and Brodie (1990) is applied. As mentioned in the previous section, if

$$\sum_{j=1}^{n} a_{ji} = \sum_{j=1}^{n} a_{ij}, \quad i = 1,,,n \tag{16}$$

the i-th row and the i-th column represent the same adjacency relationships. Thus, replacing the i-th row by zeroes does not miss any adjacency relationships. This can be proved by the nature of symmetry of an adjacency matrix **A**.

Since $\mathbf{A}^t = \mathbf{A}$, equation (16) holds for any one of the i's before changing any row. Choose the i-th row for the first trial, replacing it by zeroes, i.e., $a_{ij} = 0$ for j=1,,,n. At this point, no adjacency relationship is missing. Elements in the i-th column represent the same

adjacency relationship as the i-th row did. After changing the i-th row, if there exists a k-th row and a k-th column ($k \neq i$) that satisfy equation (16), we know that $a_{ik} = a_{ki} = 0$, i.e., the i-th unit and the k-th unit are not adjacent to each other. If they are adjacent, replacing the i-th row by zeroes subtracts one from the sum of the elements in the k-th column, but not in the k-th row, resulting in violation of equation (16). If such a k-th row-and-column exists, replacing the k-th row by zeroes also does not miss any adjacency relationship. As long as such a row-and-column exists satisfying equation (16), we can replace the corresponding row by zeroes without missing any adjacency relationship.

This adjacency matrix without redundant rows (called a reduced adjacency matrix (RAM)) can also be applied to the above proposition. In other words, this proves the sufficiency of RAM to generate adjacency constraints. The number of constraints will be less than or equal to the number of reference units.

Reduction of a redundant element, a_{ij} or a_{ji}, from the RAM, where $a_{ij} = a_{ji} = 1$, is accomplished by changing such an element above or below the principal diagonal of the RAM to zero (called a reduced triangular adjacency matrix (RTAM)). The proposition is also effective for RTAM.

As a result, we can use the proposition to form four different adjacency constraint formulations based on the

original adjacency matrix, TAM, RAM, and RTAM. To demonstrate the derivation of adjacency constraints, the example in Figure 2.1 is used.

The adjacency matrix becomes:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

A control vector $\mathbf{X}$ has a (5 x 1) dimension:

$$\mathbf{X} = (X_1, X_2, X_3, X_4, X_5)^t \tag{35}$$

If we decide to select unit 1 and unit 5, then a control vector becomes:

$$\mathbf{X} = (1,0,0,0,1)^t$$

An adjacency vector $\mathbf{V}$ is:

$$\mathbf{V} = \mathbf{A}\cdot\mathbf{X} = \begin{bmatrix} X_2 + X_4 \\ X_1 + X_3 + X_4 \\ X_2 + X_4 + X_5 \\ X_1 + X_2 + X_3 + X_5 \\ X_3 + X_4 \end{bmatrix} \tag{36}$$

For instance, since the fourth unit has four surrounding units, 1, 2, 3 and 5, then the following equality provides all adjacency constraints for the fourth unit:

$$X_4 \cdot (X_1 + X_2 + X_3 + X_5) = 0 \tag{37}$$

By converting a nonlinear constraint (37) into a linear constraint, the following linear constraint can be obtained:

$$\begin{aligned} & X_1 + X_2 + X_3 + X_5 \le 4(1 - X_4) \\ \text{or} \quad & \\ & 4X_4 + X_1 + X_2 + X_3 + X_5 \le 4. \end{aligned} \tag{38}$$

Since

$$\mathbf{A}\cdot\mathbf{1} = (2,\ 3,\ 3,\ 4,\ 2)^t \tag{39}$$

adjacency constraints based on the original adjacency matrix are:

$$\mathbf{M}\cdot\mathbf{X} = \begin{bmatrix} 2 & 1 & 0 & 1 & 0 \\ 1 & 3 & 1 & 1 & 0 \\ 0 & 1 & 3 & 1 & 1 \\ 1 & 1 & 1 & 4 & 1 \\ 0 & 0 & 1 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix} \leq \begin{bmatrix} 2 \\ 3 \\ 3 \\ 4 \\ 2 \end{bmatrix} = \mathbf{A}\cdot\mathbf{1} \tag{40}$$

Since the TAM is:

$$\text{TAM} : \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

based on the TAM the adjacency constraints become:

$$\mathbf{M}\cdot\mathbf{X} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 3 & 0 \\ 0 & 0 & 1 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 1 \\ 1 \\ 3 \\ 2 \end{bmatrix} = \mathbf{A}\cdot\mathbf{1} \tag{41}$$

As for RAM, since the first and third rows of the original adjacency matrix are redundant, elements in those rows are replaced by zeroes. The following matrix is obtained:

$$\text{RAM} : \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Based on the RAM, we have:

$$\mathbf{M \cdot X} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 3 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 4 & 1 \\ 0 & 0 & 1 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 3 \\ 0 \\ 4 \\ 2 \end{bmatrix} = \mathbf{A \cdot 1} \qquad (42)$$

Reducing redundant elements from the RAM, the following adjacency constraints based on the RTAM are obtained:

$$\text{RTAM : } \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

$$\mathbf{M \cdot X} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 3 & 0 \\ 0 & 0 & 1 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \end{bmatrix} \leq \begin{bmatrix} 0 \\ 2 \\ 0 \\ 3 \\ 2 \end{bmatrix} = \mathbf{A \cdot 1} \qquad (43)$$

The relationships in (40), (41), (42) and (43) include redundant control variables and redundant constraints, but each is a true set of adjacency constraints. The choice of constraints is dependent upon the user. The first constraint formulation (40) based on the original adjacency matrix does not require any calculation to further reduce the number of constraints and control variables in the constraints. The second formulation (41) based on TAM requires a little calculation to eliminate redundant elements from the original adjacency matrix. The number of control variables in each constraint is less than or equal to the number in the first formulation (40). The number of

constraints is always less than the number of constraints in the first formulation (40) by one. The third formulation (42) based on RAM requires calculation to eliminate redundant rows. The number of constraints is less than or equal to that obtained in the second formulation (41), while the number of control variables in each constraint is the same or greater. The last formulation (43) based on RTAM requires two operations to reduce redundant elements and rows. The result is that the number of constraints is the same as (42), but the number of control variables may be reduced. Adjacency constraints derived from the last formulation (43), therefore have a lesser number of control variables in constraints and a lesser number of constraints than the three other formulations.

Table 2.3 shows adjacency constraints derived for the example in Figure 2.1 from the proposed algorithm based on each adjacency matrix in addition to constraints resulting from application of the conventional, the M-K-J and the T-B algorithms. The number of adjacency constraints for the conventional algorithm, the M-K-J algorithm, the T-B algorithm are 7, 2, and 3, respectively. The number of adjacency constraints based on the original adjacency matrix, TAM, RAM and RTAM applied to the proposition are 5, 4, 3 and 3, respectively.

Table 2.3.Adjacency constraints derived from each algorithm

ALGORITHM	ADJACENCY CONSTRAINTS
CONVENTIONAL ALGORITHM	X1+X2<=1 X1+X4<=1 X2+X3<=1 X2+X4<=1 X3+X4<=1 X3+X5<=1 X4+X5<=1
M-K-J ALGORITHM	3X2+3X4+X1+X3<=3 X3+X4+X5<=1
T-B ALGORITHM	6X2+3X1+4X4+2X3<=6 7X4+3X1+5X2+4X3+2X5<=7 3X5+2X4+2X3<=3
PROPOSED ALGORITHM BY THE ORIGINAL ADJACENCY MATRIX (OAM)	2X1+X2+X4<=2 X1+3X2+X3+X4<=3 X2+3X3+X4+X5<=3 X1+X2+X3+4X4+X5<=4 X3+X4+2X5<=2
PROPOSED ALGORITHM BY THE TRIANGULAR ADJACENCY MATRIX (TAM)	X1+X2<=1 X2+X3<=1 X1+X2+X3+3X4<=3 X3+X4+2X5<=2
PROPOSED ALGORITHM BY THE REDUCED ADJACENCY MATRIX (RAM)	X1+3X2+X3+X4<=3 X1+X2+X3+4X4+X5<=4 X3+X4+2X5<=2
PROPOSED ALGORITHM BY THE REDUCED TRIANGULAR ADJACENCY MATRIX (RTAM)	X1+2X2+X3<=2 X1+X2+X3+3X4<=3 X3+X4+2X5<=2

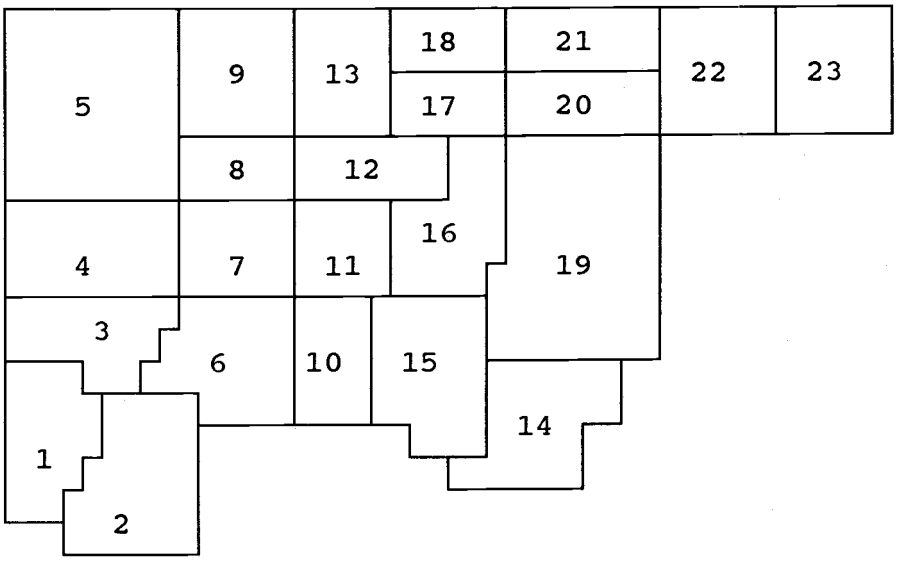
COMPARISON OF EFFICIENCY OF ALGORITHMS

In this section, two adjacency maps are introduced for further comparison of both the effort in deriving constraints and the number of constraints that results for seven different formulations. Then a one-period harvest scheduling problem is solved for each of the seven different formulations in order to investigate the influence of different formulations on computational effort.

The first example (Figure 2.3-a) has 23 units with 12 triplets and 13 pairs of inequalities. The second example (Figure 2.3-b) has 20 units with 1 triplet and 28 pairs of inequalities.

Applying the conventional algorithm to the first example, results in 41 constraints. The M-K-J algorithm yields 25 type 1 inequalities and 121 possible combinations of type 1 inequalities. The final number of constraints becomes 12. Application of the T-B algorithm results in the identification of 14 reference areas, implying that the number of final constraints is 14. The total number of entering units in the final constraints is 53, requiring development of 88 identical inequalities to achieve the final constraints. The proposed algorithm yields 23 constraints based on the original adjacency matrix, 22 constraints based on TAM, 14 constraints based on RAM, and 14 constraints based on RTAM. Since RAM and RTAM utilize

a)



b)

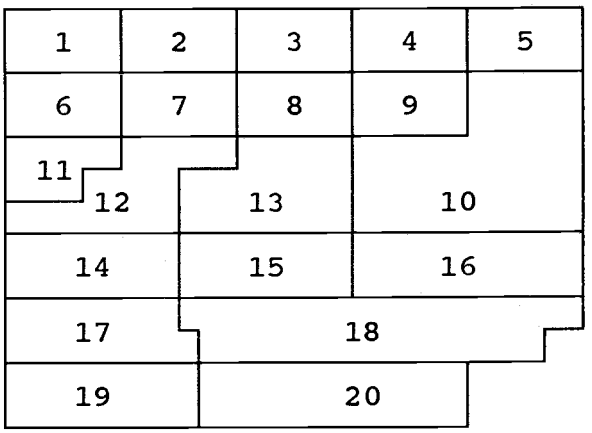


Figure 2.3. Two example structures
a) 23 unit problem
b) 20 unit problem

Procedure 1 in the T-B algorithm, the numbers of constraints are the same as for the T-B algorithm. However, the number of control variables in the constraints based on RTAM is less than for the T-B algorithm, while the number based on RAM is the same as for the T-B algorithm. Final constraint sets for all algorithms are shown in Table 2.4.

For the second example (Figure 2.3-b), the conventional algorithm yields 31 constraints. Application of the M-K-J algorithm results in the creation of 29 type 1 inequalities, and 232 possible combinations of type 1 inequalities. The final number of constraints is 10. The T-B algorithm yields 10 final constraints, 34 entering units, and 36 identical inequalities. The proposed algorithm based on the original adjacency matrix, TAM, RAM and RTAM provides 20, 19, 10 and 10 constraints, respectively. The final constraint sets for each algorithm are presented in Table 2.5.

The degree of effort required to generate the final adjacency constraints is expressed in Table 2.6. The degree of effort is a subjective rating which is based on the time for completing procedures by hand. The conventional algorithm has the largest number of constraints, but the degree of effort is low. This is due to the fact that when an adjacency matrix is determined, no further calculation is required. In contrast, the M-K-J algorithm results in the smallest number of constraints but requires a high degree of effort, resulting from generating type 1 inequalities and

Table 2.4. Adjacency constraints for the 23 unit problem

CONV.	M-K-J ALGORITHM	ALGORITHM BY OAM	ALGORITHM BY TAM
X1+X2<=1	2X20+2X21+3X22+X23<=3	X3+X2+2X1<=2	X3+X2+2X1<=2
X1+X3<=1	X1+3X2+3X3+X6<=3	X6+X3+3X2+X1<=3	X6+X3+2X2<=2
X2+X3<=1	X14+3X15+3X19+X16<=3	X6+X4+4X3+X2+X1<=4	X6+X4+2X3<=2
X2+X6<=1	2X13+2X17+3X18+X21<=3	X7+X5+3X4+X3<=3	X7+X5+2X4<=2
X3+X4<=1	2X5+2X8+3X9+X13<=3	X9+X8+3X5+X4<=3	X9+X8+2X5<=2
X3+X6<=1	X6+3X10+2X11+2X15<=3	X10+X7+4X6+X3+X2<=4	X10+X7+2X6<=2
X4+X5<=1	X3+5X4+X5+X7<=5	X11+X8+4X7+X6+X4<=4	X11+X8+2X7<=2
X4+X7<=1	3X11+X12+X15+3X16<=3	X12+X9+4X8+X7+X5<=4	X12+X9+2X8<=2
X5+X8<=1	3X12+X13+X16+3X17<=3	X13+3X9+X8+X5<=3	X13+X9<=1
X5+X9<=1	X17+X19+3X20<=3	X15+X11+3X10+X6<=3	X15+X11+2X10<=2
X6+X7<=1	X7+3X8+X12<=3	X16+X15+X12+5X11+X10+X7<=5	X16+X15+X12+3X11<=3
X6+X10<=1	X6+3X7+X11<=3	X17+X16+X13+5X12+X11+X8<=5	X17+X16+X13+3X12<=3
X7+X8<=1		X18+X17+4X13+X12+X9<=4	X18+X17+2X13<=2
X7+X11<=1		X19+X15+2X14<=2	X19+X15+2X14<=2
X8+X9<=1		X19+X16+5X15+X14+X11+X10<=5	X19+X16+2X15<=2
X8+X12<=1		X19+X17+5X16+X15+X12+X11<=5	X19+X17+2X16<=2
X9+X13<=1		X20+X18+5X17+X16+X13+X12<=5	X20+X18+2X17<=2
X10+X11<=1		X21+3X18+X17+X13<=3	X21+X18<=1
X10+X15<=1		X20+4X19+X16+X15+X14<=4	X20+X19<=1
X11+X12<=1		X22+X21+4X20+X19+X17<=4	X22+X21+2X20<=2
X11+X15<=1		X22+3X21+X20+X18<=3	X22+X21<=1
X11+X16<=1		X23+3X22+X21+X20<=3	X23+X22<=1
X12+X13<=1		X23+X22<=1	
X12+X15<=1	12	23	22
X12+X17<=1	T-B ALGORITHM	ALGORITHM BY RAM	ALGORITHM BY RTAM
X13+X17<=1	6X2+4X1+4X3+3X6<=6	X6+X3+3X2+X1<=3	X1+3X2+X3+X6<=3
X13+X18<=1	6X3+3X1+4X2+X4+2X6<=6	X6+X4+4X3+X2+X1<=4	X1+3X3+X4+X6<=3
X14+X15<=1	4X5+X4+2X8+2X9<=4	X9+X8+3X5+X4<=3	X4+3X5+X8+X9<=3
X14+X19<=1	4X7+X4+X6+X8+X11<=4	X11+X8+4X7+X6+X4<=4	X4+X6+4X7+X8+X11<=4
X15+X16<=1	4X9+2X5+2X8+X13<=4	X13+3X9+X8+X5<=3	X8+2X9+X13<=2
X15+X19<=1	4X10+X6+2X11+2X15<=4	X15+X11+3X10+X6<=3	X6+3X10+X11+X15<=3
X16+X17<=1	8X12+X8+3X11+2X13+4X16+3X17<=8	X17+X16+X13+5X12+X11+X8<=5	X8+X11+5X12+X13+X16+X17<=5
X16+X19<=1	9X15+3X10+5X11+2X14+4X16+3X19<=9	X19+X16+5X15+X14+X11+X10<=5	X11+X14+4X15+X16+X19<=4
X17+X18<=1	10X16+6X11+5X12+4X15+2X17+2X19<=10	X19+X17+5X16+X15+X12+X11<=5	X11+3X16+X17+X19<=3
X17+X20<=1	8X17+5X12+5X13+2X16+2X18+X20<=8	X20+X18+5X17+X16+X13+X12<=5	X13+3X17+X18+X20<=3
X18+X21<=1	4X18+2X13+2X17+X21<=4	X21+3X18+X17+X13<=3	X13+2X18+X21<=2
X19+X20<=1	6X19+3X14+4X15+2X16+X20<=6	X20+4X19+X16+X15+X14<=4	X14+2X19+X20<=2
X20+X21<=1	4X21+X18+2X20+2X22<=4	X22+3X21+X20+X18<=3	X20+2X21+X22<=2
X20+X22<=1	4X22+2X20+2X21+X23<=4	X23+3X22+X21+X20<=3	X20+2X22+X23<=2
X21+X22<=1			
X22+X23<=1			
41*	14	14	14

*: Number of constraints
CONV.:the conventional pairwise formulation
OAM: the original adjacency matrix
TAM: the triangular adjacency matrix
RAM: the reduced adjacency matrix
RTAM: the reduced triangular adjacency matrix

Table 2.5.Adjacency constraints for the 20 unit problem

CONV.	M-K-J ALGORITHM	ALGORITHM BY OAM	ALGORITHM BY TAM
X1+X2<=1	17X+3X19+X20<=3	X6+X2+2X1<=2	X6+X2+2X1<=2
X1+X6<=1	X1+5X2+X3+X7<=5	X7+X3+3X2+X1<=3	X7+X3+2X2<=2
X2+X3<=1	X1+5X6+X7+X11<=5	X8+X4+3X3+X2<=3	X8+X4+2X3<=2
X2+X7<=1	X3+5X4+X5+X9<=5	X9+X5+3X4+X3<=3	X9+X5+2X4<=2
X3+X4<=1	3X15+3X16+5X18+X17+X20<=5	X10+2X5+X4<=2	X10+X5<=1
X3+X8<=1	X12+5X14+X15+X17<=5	X11+X7+3X6+X1<=3	X11+X7+2X6<=2
X4+X5<=1	X5+7X10+X9+X13+X16<=7	X12+X8+4X7+X6+X2<=4	X12+X8+2X7<=2
X4+X9<=1	X7+5X12+X11+X13<=5	X13+X9+4X8+X7+X3<=4	X13+X9+2X8<=2
X5+X10<=1	X3+7X8+X7+X9+X13<=7	X10+3X9+X8+X4<=3	X10+X9<=1
X6+X7<=1	X13+X15<=1	X16+X13+4X10+X9+X5<=4	X16+X13+2X10<=2
X6+X11<=1		X12+2X11+X6<=2	X12+X11<=1
X7+X8<=1		X14+X13+4X12+X11+X7<=4	X14+X13+2X12<=2
X7+X12<=1		X15+4X13+X12+X10+X8<=4	X15+X13<=1
X8+X9<=1		X17+X15+3X14+X12<=3	X17+X15+2X14<=2
X8+X13<=1		X18+X16+4X15+X14+X13<=4	X18+X16+2X15<=2
X9+X10<=1		X18+3X16+X15+X10<=3	X18+X16<=1
X10+X13<=1		X19+X18+3X17+X14<=3	X19+X18+2X17<=2
X10+X16<=1		X20+4X18+X17+X16+X15<=4	X20+X18<=1
X11+X12<=1		X20+2X19+X17<=2	X20+X19<=1
X12+X13<=1		2X20+X19+X18<=2	
X12+X14<=1	10	20	19
X13+X15<=1	T-B ALGORITHM	ALGORITHM BY RAM	ALGORITHM BY RTAM
X14+X15<=1	3X2+X1+X3+X7<=3	X7+X3+3X2+X1<=3	X1+3X2+X3+X7<=3
X14+X17<=1	3X4+X3+X5+X9<=3	X9+X5+3X4+X3<=3	X3+3X4+X5+X9<=3
X15+X16<=1	3X6+X1+X7+X11<=3	X11+X7+3X6+X1<=3	X1+3X6+X7+X11<=3
X15+X18<=1	4X8+X3+X7+X9+X13<=4	X13+X9+4X8+X7+X3<=4	X3+X7+4X8+X9+X13<=4
X16+X18<=1	4X10+X5+X9+X13+X16<=4	X16+X13+4X10+X9+X5<=4	X5+X9+4X10+X13+X16<=4
X17+X18<=1	4X12+X7+X11+X13+X14<=4	X14+X13+4X12+X11+X7<=4	X7+X11+4X12+X13+X14<=4
X17+X19<=1	5X15+X13+X14+2X16+2X18<=5	X18+X16+4X15+X14+X13<=4	X13+X14+4X15+X16+X18<=4
X18+X20<=1	3X17+X14+X18+X19<=3	X19+X18+3X17+X14<=3	X14+3X17+X18+X19<=3
X19+X20<=1	4X18+2X15+2X16+X17+X20<=4	X20+4X18+X17+X16+X15<=4	X16+2X18+X20<=2
	2X20+X18+X19<=2	2X20+X19+X18<=2	X19+X20<=1
31*	10	10	10

*: Number of constraints
CONV.:the conventional pairwise formulation
OAM: the original adjacency matrix
TAM: the triangular adjacency matrix
RAM: the reduced adjacency matrix
RTAM: the reduced triangular adjacency matrix

Table 2.6. A comparison of the efficiency of the algorithms

Algorithms	Problem	Number of Constraints	The Degree of Effort
Conventional Algorithm	23 Units	41	Low (no further calculation)
	20 Units	31	Low (no further calculation)
M-K-J Algorithm	23 Units	12	High (121 combinations of Type 2)
	20 Units	10	High (232 combinations of Type 2)
T-B Algorithm	23 Units	14	Moderate (88 identical inequalities)
	20 Units	10	Moderate (36 identical inequalities)
Proposed Algorithm By OAM	23 Units	23	Low(no arrangement of adjacency matrix)
	20 Units	20	Low(no arrangement of adjacency matrix)
Proposed Algorithm By TAM	23 Units	22	Low (reduction of redundant elements)
	20 Units	19	Low (reduction of redundant elements)
Proposed Algorithm By RAM	23 Units	14	Low (reduction of redundant rows)
	20 Units	10	Low (reduction of redundant rows)
Proposed Algorithm By RTAM	23 Units	14	Low (reduction of rows and elements)
	20 Units	10	Low (reduction of rows and elements)

OAM : the original adjacency matrix, TAM : the triangular adjacency matrix,
RAM : the reduced adjacency matrix, RTAM : the reduced triangular adjacency matrix

all possible type 2 inequalities, and then selecting the final constraints. As for the T-B algorithm, it yields a reduced number of constraints with a moderate degree of effort due to the heuristic of Procedure 2. The proposed algorithm yields a reduced number of constraints, and the degree of effort is low. Once an adjacency matrix is determined, only the elementary algebra is required, i.e., eliminating redundant rows and elements from an adjacency matrix.

Although the number of constraints is dependent upon the structure of the problem, the M-K-J algorithm seems to result in fewer constraints as the number of triplets increases. From the number of all possible combinations of type 1 inequalities for the two examples in Figure 2.3, it is noticeable that the more triplets, the less combinations of type 1 inequalities. The T-B algorithm, on the other hand, requires more identical inequalities with an increased number of triplets.

It is interesting to note the relationship between the numbers of constraints among different algorithms. Our conclusion is that the number of constraints resulting from the M-K-J algorithm might be less than or equal to the number of constraints from the T-B algorithm and the proposed algorithm. This is due to the fact that each constraint generated by the proposed algorithm can be formulated in one constraint by the M-K-J algorithm. To

prove this, consider the general constraint (29) generated by the proposed algorithm in the previous section:

$$\sum_{j \epsilon \mathbf{S}_i} X_j \leq r_i \cdot [1 - X_i]$$

or

$$r_i \cdot X_i + \sum_{j \epsilon \mathbf{S}_i} X_j \leq r_i \qquad (29)$$

This constraint can be disaggregated into:

$$X_i + X_j \leq 1, \quad \text{for all } j \epsilon \mathbf{S}_i \qquad (44)$$

The j-th unit ($j \epsilon \mathbf{S}_i$) has the i-th unit in common, and the above equation (44) satisfies the first condition of the M-K-J algorithm for type 2 inequalities:

$$\sum_{i \epsilon E} X_i + X_j \leq 1, \; j \epsilon P \; (= \mathbf{S}_i) \qquad (7)$$

Therefore we can formulate a type 2 inequality as follows:

$$(2r_i - 1) \cdot X_i + \sum_{j \epsilon P} X_j \leq (2r_i - 1) \qquad (45)$$

This implies that there exist such rules for the M-K-J algorithm that result in the same number of constraints as that of the proposed algorithm. If the heuristic rules of the M-K-J algorithm are more efficient than such rules, the number of adjacency constraints formulated by the M-K-J algorithm could be less than or equal to that of the other algorithms.

In considering the T-B algorithm, it is sufficient to penalize coefficients of only reference units by the number of entering units (adjacent units) so that the T-B algorithm formulates the same adjacency constraints as the proposed algorithm based on RAM. Penalization of entering units

considers adjacency relationships among entering units, which are already represented by relationships with other reference units.

To evaluate the influence of different formulations on the computational task of solving the resultant integer programming problem, LINGO/386 (Language for INteractive General Optimization) by Cunningham and Schrage (1989) was used. Like other commercial softwares, LINGO/386 incorporates the branch-and-bound algorithm. The branch-and-bound algorithm was developed by Land and Doig (1960) and Dakin (1965). The algorithm starts by solving the integer programming problem without any integer restriction on integer variables. A tree search is then utilized by imposing bounds on integer variables which do not attain integer values. Once all integer variables attain integer values, one feasible solution is obtained. When the tree search is completed, the best feasible solution is identified.

The computational task for seven different formulations is evaluated based on the number of pivots and the real time in seconds. A pivot is a transformation which corresponds to a step in the Gaussian elimination technique for solving linear equations (Nemhauser and Wolsey 1988). The detail of pivot operations can be found in operations research textbooks, such as Bazaraa and Jarvis (1977) and Winston (1987).

The objective of the problem is to maximize the sum of returns from each dichotomous control variable. Table 2.7 shows the return from each unit for the two examples. Both problems are solved over a single period.

Table 2.8 shows results from the seven different formulations. Linear programming relaxed solutions and the final integer solutions are provided along with the corresponding number of pivots and computational time. For both examples, the integer solutions from all formulations are the same. In terms of the number of pivots to obtain a linear programming relaxed solution, we have the following relationship among formulations:

23 unit problem:

T-B < M-K-J = RAM = RTAM < OAM = TAM < Conv.

20 unit problem:

T-B < RAM = RTAM < M-K-J < TAM <OAM < Conv.

where the notation is defined by:

Conv.: the conventional algorithm,

M-K-J: the M-K-J algorithm,

T-B : the T-B algorithm,

OAM : the proposed algorithm based on the original adjacency matrix,

TAM : the proposed algorithm based on TAM,

RAM : the proposed algorithm based on RAM,

RTAM : the proposed algorithm based on RTAM.

Based on the objective value of the linear programming

Table 2.7.Return from each unit for both 23 and 20 unit problems

UNIT #	23 UNIT PROBLEM	20 UNIT PROBLEM
1	*1987.5	1546.3
2	1359.8	1161.0
3	1048.8	474.7
4	2143.4	346.5
5	1466.4	668.9
6	1131.1	488.2
7	615.0	1026.2
8	448.9	702.1
9	360.5	831.4
10	1247.1	568.8
11	853.2	1776.0
12	658.1	1333.5
13	2189.3	1852.5
14	1643.8	1390.9
15	1173.0	1469.8
16	744.4	1103.5
17	543.4	1362.6
18	436.4	1023.1
19	744.4	1156.2
20	543.4	791.0
21	436.4	*****
22	1351.0	*****
23	924.3	*****

*: dollar ($)

Table 2.8. Results of computational comparisons among algorithms

ALGORITHM	Example Problems	LP relaxed solution		Final IP solution			
		Objective Value	#of Pivots	Objective Value	# of Pivots	TIME(second)	TIME/PIVOT(second)
CONVENTIONAL ALGORITHM	23 UNIT PROBLEM	12165.2	33	11872.1	54	6.97	0.129
	20 UNIT PROBLEM	11826.6	27	11826.6	27	4.72	0.175
M-K-J ALGORITHM	23 UNIT PROBLEM	14689.4	20	11872.1	2489	56.46	0.023
	20 UNIT PROBLEM	14696.3	20	11826.6	1150	25.76	0.022
T-B ALGORITHM	23 UNIT PROBLEM	14256.8	18	11872.1	2601	59.31	0.023
	20 UNIT PROBLEM	13206.0	17	11826.6	584	12.74	0.022
PROPOSED ALGORITHM BY THE ORIGINAL ADJACENCY MATRIX (OAM)	23 UNIT PROBLEM	13345.0	25	11872.1	367	14.88	0.041
	20 UNIT PROBLEM	12433.3	25	11826.6	113	6.81	0.060
PROPOSED ALGORITHM BY THE TRIANGULAR ADJACENCY MATRIX (TAM)	23 UNIT PROBLEM	13510.9	25	11872.1	322	10.54	0.033
	20 UNIT PROBLEM	12160.2	23	11826.6	44	4.77	0.108
PROPOSED ALGORITHM BY THE REDUCED ADJACENCY MATRIX (RAM)	23 UNIT PROBLEM	15657.7	20	11872.1	2864	70.52	0.025
	20 UNIT PROBLEM	13279.5	17	11826.6	725	15.54	0.021
PROPOSED ALGORITHM BY THE REDUCED TRIANGULAR ADJACENCY MATRIX (RTAM)	23 UNIT PROBLEM	15630.7	20	11872.1	1938	39.71	0.020
	20 UNIT PROBLEM	12984.7	23	11826.6	533	11.09	0.021

relaxed solution, however, we have:

23 unit problem:

Conv. < OAM < TAM < T-B < M-K-J < RTAM < RAM

20 unit problem:

Conv. < TAM < OAM < RAM < T-B < RTAM < M-K-J.

As for the final integer solution, we have different orders among formulations from the above results. In terms of the number of pivots:

23 unit problem:

Conv. < TAM < OAM < RTAM < M-K-J < T-B < RAM
(1) (6) (6.8) (35.9) (46.1)(48.2) (53)

20 unit problem:

Conv. < TAM < OAM < RTAM < T-B < RAM < M-K-J
(1) (1.6) (4.2) (19.7) (21.6)(26.9)(42.6)

The value in the parenthesis indicates the ratio of the number of pivots to the number of pivots for the conventional algorithm. Williams (1974) defines formulations, such as the conventional, for which numerous pivots are required in the linear programming relaxed solution and a low value for its objective function, as "tight." Tighter formulations will tend to be more efficiently solved in the integer phase. For instance, the proposed algorithm based on RAM takes 53 times more pivots than the conventional algorithm for the 23 unit problem, and the M-K-J algorithm takes 42.6 times more than the conventional algorithm for the 20 unit problem. On the basis of the computational time, we have the same order as

the above with different ratios:

23 unit problem:

Conv.	<	TAM	<	OAM	<	RTAM	<	M-K-J	<	T-B	<	RAM
(1)		(1.5)		(2.1)		(5.7)		(8.1)		(8.5)		(10.1)

20 unit problem:

Conv.	<	TAM	<	OAM	<	RTAM	<	T-B	<	RAM	<	M-K-J
(1)		(1.0)		(1.4)		(2.3)		(2.7)		(3.3)		(5.5)

From the above results, it is possible to divide the seven different formulations into three groups:

{Conv.}, {OAM, TAM}, {M-K-J, T-B, RAM, RTAM}.

In terms of the number of pivots and computational time, the first group is better than the other two groups, and the second group is better than the third group, while in terms of the number of adjacency constraints and computational time per pivot, the opposite relationship is true.

CONCLUSIONS

In this paper, an analytical algorithm to generate adjacency constraints is proposed. The main advantage of the proposed algorithm over the other two heuristic algorithms is that the proposed algorithm is theoretically proved to be correct and it is easy to create constraints. The advantage over the conventional algorithm is the small number of constraints.

Although the M-K-J algorithm provides fewer adjacency constraints, a much larger effort is required. The M-K-J algorithm efficiently transforms a group of type 1 inequalities into one type 2 inequality. However, the rest of the M-K-J procedures remain inefficient in terms of the effort to determine the final constraints.

Procedure 1 in the T-B algorithm is theoretically proved to be correct in reducing redundant constraints and sufficient to generate constraints. Because of this, the T-B algorithm has some advantage in effort over the M-K-J algorithm. Procedure 2, on the other hand, is heuristic and can cause difficulty for understanding and explaining the procedures. This might be the reason that Torres and Brodie (1990) mentioned the possibility of violating adjacency relationships in using the reference units to create adjacency constraints.

One of the characteristics of both the M-K-J algorithm

and the T-B algorithm is that both were derived from the two dimensional map, where only the three patterns of adjacency relationship in Figure 2.2 can be drawn, while the proposed algorithm was not. Since the proposed algorithm was derived by means of matrix algebra, any adjacency matrix can be applied. For instance, suppose our problem is extended to the multiperiod problem. Additional constraints may be that only one treatment (decision) is made for each unit. This constraint is easily included in the adjacency matrix, since each decision for one unit has adjacency relationships with other decisions. Once the adjacency matrix is obtained, the rest of the work is easy.

The M-K-J algorithm creates the smallest number of adjacency constraints, followed by the T-B algorithm and the proposed algorithm based on both RAM and RTAM, followed by the proposed algorithm based on TAM, the original adjacency matrix, and the conventional algorithm. However, from our experiments (even though not exhaustive), we can conclude that it is not always true that the less the number of adjacency constraints, the less the computational time for integer programming problems. It may be true that the less the number of adjacency constraints, the less the computational time per pivot.

It is worth noting the following three points Williams (1974) pointed out from his experiments in integer programming:

a) Ingenious formulations to make a model more compact (less restraints) are usually not as good (efficient in integer solution) as less sophisticated formulations.

b) It is desirable to make an integer programming problem as tight as possible in a continuous sense.

c) The number of iterations taken to reach the continuous optimum (linear programming relaxed solution) is usually greater in the tighter formulation.

Our experiments support the above points. Of all algorithms, the conventional algorithm is revealed to be the tightest formulation, followed by the proposed algorithm based on the original adjacency matrix and TAM.

Applying the branch-and-bound algorithm for solving a single period harvest scheduling problem, the conventional algorithm is the way to go as it appears to offer the easiest formulation to solve. For large problems where the number of constraints/variables exceeds the capabilities of the solution system being utilized, an approach such as the one outlined here is useful. In other circumstances where the use of the branch-and-bound algorithm is impractical in terms of computational task, reduction of constraints plays an important role in constructing a heuristic technique to solve the problem. The composite relaxation method presented by Torres et al. (1990) is such a heuristic.

LITERATURE CITED

Bazaraa, M.S. and J.J. Jarvis. 1977. Linear programming and network flows. John Wiley & Sons Inc., New York. 565p.

Cunningham, K. and L. Schrage. 1989. The LINGO modeling language. LINDO Systems Inc., Chicago. 93p.

Dakin, R.J. 1965. A tree search algorithm for mixed integer programming problems. Computer J., 8:250-255.

Gross, T.E. and D.P. Dykstra. 1988. Harvest Scheduling with nonadjacency constraints. In Proceedings, Society of American Foresters National Convention, Oct. 16-19, 1988. Wash. D.C. pp.310-315.

Gross, T.E. 1989. Use of graph theory to analyze constraints on the Juxtaposition of timber stands. M.S. thesis. School of Forestry, Northern Arizona University, Flagstaff, AR. 153p.

Hokans, R.H. 1983. Evaluating spatial feasibility of harvest schedules with simulated stand-selection decisions. J. For., 81:601-603, 613.

Land, A.H. and A.G. Doig. 1960. An automatic method for solving discrete programming problems. Econometrica, 28:497-520.

May, K.O. 1965. The origin of the four-color conjecture. Isis, 56:346-348.

Mealey, S.P., J.F. Lipscomb, and K.N. Johnson. 1982. Solving the habitat dispersion problem in forest planning.

Trans. N. Amer. Wildl. Natur. Resour. Conf., 47:142-153.

Meneghin, B.J., M.W.Kirby, and J.G.Jones. 1988. An algorithm for writing adjacency constraints efficiently in linear programming models. The 1988 Symp. on System Analysis in Forest Resources, March 29 - April 1 1988. Gen. Tech. Rep. Rm-161:46-53, Rocky Mtn. For. & Ran. Exp. Stn., Ft. Collins, Colorado.

Nelson, J., J.D. Brodie, and J. Sessions. 1988. Integrating short term spatially feasible harvest plans with long term harvest schedules using Monte-Carlo integer programming and linear programming. The 1988 Symp. on System Analysis in Forest Resources, March 29 - April 1 1988. Gen. Tech. Rep. Rm-161:224-229, Rocky Mtn. For. & Ran. Exp. Stn., Ft. Collins, Colorado.

Nemhauser, G.L. and L.A. Wolsey. 1988. Integer and combinatorial optimization. John Wiley & Sons Inc., New York. 763p.

Sessions, J. and J.B. Sessions. 1988. SNAP - a scheduling and network analysis program for tactical harvest planning. In Proceedings of International Mountain Logging and Pacific Northwest Skyline Symp., Dec. 12-16 1988. Oregon State Univ., Corvallis, OR. pp.71-75.

Thompson, E.F., B.G. Halterman, T.J. Lyon, and R.L. Miller. 1973. Integrating timber and wildlife management planning. Forestry Chron., Dec. 1973:247-250.

Torres R., J.M. and J.D.Brodie. 1990. Adjacency constraints in harvest scheduling: an aggregation heuristic. Can. J. For. Res. In press.

Torres R., J.M., J.D. Brodie, and J. Sessions. 1990. The use of relaxation to solve the habitat dispersion problem. In journal review.

Williams, H.P. 1974. Experiments in the formulation of integer programming problems. In: M.L. Balinski, ed., Math. Prog. Study No.2 (Approaches to Integer Programming, North-Holland, Amsterdam):82-114.

Winston, W.L. 1987. Operations research: applications and algorithms. Duxbury Press, Boston. 1025p.

Chapter 3

A COMPARISON OF APPROACHES FOR INTEGRATING SPATIAL CONCERNS INTO HARVEST SCHEDULING

by

Atsushi Yoshimoto

J. Douglas Brodie

and

Juan M. Torres R.

ABSTRACT

Due to capability limitations of the integer programming solution technique, an alternative heuristic algorithm was developed to solve spatially constrained area-based harvest scheduling problems. The technique utilizes random ordering heuristic optimization and the PATH algorithm adapted from stand level optimization. Employing the proposed algorithm, a harvest scheduling system was constructed. The performance of the proposed algorithm is presented in comparison to the branch-and-bound algorithm and the composite relaxation algorithm. Solutions found for the sample problems by the proposed algorithm are stable in terms of the objective value, and have harvest flow

fluctuation much less than 5 %. The proposed algorithm yields better solutions for the 3 or more period problems than those of the branch-and-bound algorithm using 100,000 iterations as the limit for iteration. The new algorithm also outperforms the composite relaxation algorithm. For the 1- and 2- period problems, the solutions by the proposed algorithm can produce an objective value with deviation less than 2 % from the optimal solution. From the upper bound of the corresponding linear programming relaxation, the proposed algorithm provides solutions with the objective value within 7 % for 3 or more period problems. The advantage of the proposed algorithm results from partitioning the problem period by period using the PATH algorithm, and respecification of the objective function of the subproblem by minimizing the flow fluctuation, leading to a final solution with small flow fluctuation and high objective value.

INTRODUCTION

The determination of which harvest units should be harvested, by what method, and when, is the main issue in the harvest scheduling problem. If no constraints are considered, optimization of each individual harvest unit over the time horizon (stand level optimization) constructs an optimal harvest schedule for the whole target forest. In such an unconstrained harvest scheduling problem, for example, the objective becomes to search for the optimal rotation age and optimal thinning regime for each stand, leading to an optimal harvest schedule.

Difficulty arises when harvest units are combined to meet a physical or economic objective function subject to a range of multiple-use resource constraints (forest level optimization). Constraints generally deal with harvest flows, habitat specifications and other societal concerns.

Mathematical programming techniques have been applied to not only stand level optimization but also forest level optimization. Among these techniques, a fundamental methodology, linear programming (LP), plays an important role in optimization of harvest scheduling problems. LP is a widely used optimization technique and one of the practical tools devised for complex decision-making problems that optimally allocates limited resources to competing activities. Timber RAM (Resource Allocation Model) (Navon

1971) and FORPLAN (FORest PLANing) (Johnson and Stuart 1987) are widely used LP based models for forestry problems.

To solve LP problems, the simplex algorithm developed by Dantzig (1951) has been the most successfully applied. The simplex algorithm involves a procedure that progresses from one basic feasible solution to another basic feasible solution so that each iteration is guaranteed to provide a better solution.

Due to the increasing number of decision variables and constraints in harvest scheduling problems, LP formulations of timber management become so large that solution by LP becomes infeasible, even with today's sophisticated software and modern computer technology. As a result, techniques that provide solutions that are near-feasible and near-optimal have become acceptable as a "good" solution with less cost and computational time. Minimally infeasible solutions can be provided for specifications that are inherently infeasible. LP would provide no solution for such specifications. Decomposition (Berck and Bible 1984) and simulation techniques (Lagrangean relaxation) (Hoganson and Rose 1984) are introduced as an alternative technique for large-scale LP problems.

As long as LP based models are utilized, the decision variables must be divisible and continuous, taking any nonnegative real value. A decision variable may not be divisible, however, if it represents a discrete type of

activity such as a "harvest"-or-"not harvest" decision, a "build a road"-or-"not build a road" decision and so on. In such a case, a decision variable is forced to be a dichotomous or zero-one variable. Introducing a dichotomous decision variable into the area-based model can prevent fractionalization of a harvest unit. Fractionalization would occur if part of a harvest unit were assigned to harvest and part to habitat protection. This can be avoided by requiring that only one prescription be selected. By doing so, a forest planner could readily implement prescriptions without disaggregating a solution from LP.

Harvest scheduling problems containing dichotomous decision variables can be formulated by mixed integer programming (MIP) or 0-1 integer programming (IP). The Integrated Resource Planning Model (IRPM) was developed by Kirby et al. (1980) as an MIP model for solving this type of scheduling problem.

A branch-and-bound algorithm with LP relaxation (B&B) is most widely applied to commercial codes for solving MIP or IP problems (Nemhauser and Wolsey 1988). This method was developed by Land and Doig (1960) and improved by Dakin (1965). The algorithm first solves an LP relaxed problem by ignoring integer restrictions. Then a new constraint, which forces the LP solution toward an integer solution, is added, leading to a new LP relaxed problem to be solved. The process continues until either a fully integer solution is

found or the solution becomes inferior to the best solution previously found or the problem is found to be infeasible with no integer solution.

Difficulties are encountered in optimizing MIP or IP problems by B&B when the number of integer variables is large. There are two strategies to handle difficulties. One is to arrange an IP formulation in order for the algorithm to find integer solutions at the earliest stage of the LP relaxation problem. Ghandforoush and Greber (1986) proposed an efficient IP formulation method which reduced the computational burden effectively.

The other approach is to use a heuristic. Zanakis and Evans (1981) suggested the use of a heuristic "optimization" procedure (in contrast to a more efficient formulation) in order to reduce the computational burden. The nature of a heuristic is such that a heuristic "optimization" algorithm does not necessarily provide an optimal solution but provides a "good" solution with reasonable computational cost. There are two different types of heuristic techniques in the forestry literature. One is a random search technique and the other is a composite relaxation technique. A random search technique utilized for an optimization problem in forest management was first presented by Bullard et al. (1985). They used a random search technique for a stand level optimization problem. For the forest level optimization problem, Sessions and Sessions (1988), O'Hara

et al. (1989) and Nelson et al. (1990) introduced a random search technique with or without modification. Using the other heuristic technique, Torres et al. (1990b) introduced a composite relaxation technique for a spatially constrained area-based harvest scheduling problem.

The objective of this paper is to propose another alternative technique for a spatially constrained area-based harvest scheduling problem based on a random search technique, then evaluate its performance in comparison to other techniques. The remainder of this paper is organized as follows. First, a spatially constrained area-based harvest scheduling problem is formulated as an IP problem. Then random search techniques by Sessions and Sessions (1988), O'Hara et al. (1989) and Nelson et al. (1990), and a composite relaxation technique by Torres et al. (1990b) are briefly reviewed. In the third section, the alternative technique is introduced based on a random search technique. In the fourth section, the model applying the proposed algorithm is developed, then the model experiment is conducted in the fifth section. The evaluation of the proposed algorithm is completed in the sixth section in comparison to the best solution of B&B and the composite relaxation technique developed by Torres et al. (1990b).

THE SPATIALLY CONSTRAINED HARVEST SCHEDULING PROBLEM

Integer Programming Formulation

The problem considered is a spatially constrained area-based harvest scheduling problem with even-flow and adjacency constraints. We formulate an IP problem where $\underline{X}$ is an (N x 1) dichotomous decision vector with N representing the number of decisions and, an element of $\underline{X}$ is defined by,

$$x_i = \begin{cases} 1 & \text{if the i-th decision is implemented} \\ 0 & \text{otherwise} \end{cases}$$

The objective is to maximize the present net worth of return from each decision,

$$Z_{IP} = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} \} \qquad (1)$$

where $\underline{C}$ is an (N x 1) coefficient vector of $\underline{X}$, representing present net worth of return and t denotes the transpose. In this paper, a decision is meant to be a certain treatment of a harvest unit, e.g., "harvest the 5-th unit at period three" as in the Model I formulation (Johnson and Stuart 1988). Even-flow constraints can be expressed by,

$$\underline{V} \cdot \underline{X} = \underline{V0} \qquad (2)$$

where $\underline{V0}$ is a (T x 1) even-flow vector, and $\underline{V}$ is a (T x N) matrix whose element v_{ik} represents harvestable volume contributed by the k-th decision at the i-th period and T is the number of periods.

Great attention has been paid to the adjacency constraint formulation. One of the most common formulations is through conventional pairwise adjacency constraints. Each pairwise adjacency constraint has the following form,

$$x_i + x_j \leq 1 \qquad \text{for all i and j (>i)} \qquad (3)$$

where the i-th and the j-th decisions cannot be achieved at the same time and these two decisions do not belong to the same harvest unit.

As for other formulations, Meneghin et al. (1988), Torres and Brodie (1990), and Yoshimoto and Brodie (1990) proposed efficient adjacency formulations in a sense that the formulation has less constraints than the conventional pairwise adjacency constraints. Unless the conventional pairwise adjacency formulation is applied, an adjacency matrix, $\underline{\mathbf{A}}$ is necessary for other formulations. This adjacency matrix consists of a_{ij} elements whose values are one if the i-th harvest unit is adjacent to the j-th unit, and zero otherwise. Using n as the number of harvest units, $\underline{\mathbf{A}}$ has a (n x n) dimension. Expanding the adjacency matrix for N decisions by replacing harvest units by decisions, $\underline{\mathbf{A}}$ has an (N x N) dimension. An element a_{ij} of $\underline{\mathbf{A}}$ is one if the i-th and j-th decisions cannot be implemented at the same time, and zero otherwise. Again, note that these two decisions don't belong to the same harvest unit.

Meneghin et al. (1988), Torres and Brodie (1990), and Yoshimoto and Brodie (1990) have recently proposed

algorithms for efficiently and sparsely specifying adjacency constraints. The Yoshimoto and Brodie (1990) approach can be systematically and analytically derived without use of heuristics, and the formulation can be proved to be correct in the sense that no adjacent harvests occur. In this paper, we will use Yoshimoto and Brodie's (1990) formulation to generate adjacency constraints for both IP and alternative solution methods. Since the conventional algorithm was shown to be most efficient for B&B, the IP examples are also formulated using this algorithm.

Following Yoshimoto and Brodie (1990) for the purpose of simplification of the formulation, adjacency constraints are formulated by,

$$\underline{\mathbf{M}} \cdot \underline{\mathbf{X}} \leq \underline{\mathbf{M0}} \qquad (4)$$

where

$$\underline{\mathbf{M0}} = \underline{\mathbf{A}} \cdot \underline{\mathbf{1}}$$

$$\underline{\mathbf{M}} = \underline{\mathbf{A}} + \underline{\mathbf{B}}$$

and $\underline{\mathbf{B}}$ is a diagonal matrix whose i-th diagonal element is the same as the i-th element of $\underline{\mathbf{M0}}$ and $\underline{\mathbf{1}}$ is a unit vector. Each element a_{ij} of $\underline{\mathbf{A}}$ is defined by,

$$a_{ij} = \begin{cases} 1 & \text{if the i-th and j-th decisions cannot be achieved together} \\ 0 & \text{otherwise} \end{cases}$$

Equation (4) can be used for the conventional pairwise formulation by changing dimension and elements of $\underline{\mathbf{M}}$ and $\underline{\mathbf{M0}}$.

The last constraints are the land accounting constraints which imply that only one decision is achieved

for one harvest unit over the time horizon in considering "no activity" as one decision for the unit. Letting D_i be a decision set for the i-th harvest unit, these land accounting constraints are expressed by,

$$\sum_{j \epsilon D_i} x_j = 1 \qquad \text{for } i = 1,,,n \tag{5}$$

Deleting the no-activity decision from the decision set, the equality in (5) becomes an inequality, i.e., less than or equal to 1. Substituting the matrix notation, equation (5) becomes;

$$\underline{\mathbf{L}} \cdot \underline{\mathbf{X}} \leq \underline{\mathbf{1}} \tag{6}$$

where $\underline{\mathbf{L}}$ is an (n x N) matrix. The i-th column vector of $\underline{\mathbf{L}}$ corresponds to the decisions for the i-th harvest unit. Any decision of $\underline{\mathbf{X}}$ for the i-th harvest unit has 1 value in the i-th column vector of $\underline{\mathbf{L}}$.

As a result, the following IP formulation is utilized,

(IP) $$Z_{IP} = \underset{\{\underline{\mathbf{X}}\}}{\text{maximize}} \{ \underline{\mathbf{C}}^t \cdot \underline{\mathbf{X}} \} \tag{1}$$

subject to

$$\underline{\mathbf{V}} \cdot \underline{\mathbf{X}} = \underline{\mathbf{V0}} \tag{2}$$

$$\underline{\mathbf{M}} \cdot \underline{\mathbf{X}} \leq \underline{\mathbf{M0}} \tag{4}$$

$$\underline{\mathbf{L}} \cdot \underline{\mathbf{X}} \leq \underline{\mathbf{1}} \tag{6}$$

$$\underline{\mathbf{X}} \epsilon \{0, 1\}$$

It is interesting to mention that an adjacency matrix $\underline{\mathbf{A}}$ can be augmented to include the relationship among all decision variables. In other words, the augmented (N x N) adjacency matrix $\underline{\mathbf{A}}'$ includes not only the usual two

dimensional adjacency relationship among harvest units described by constraints (4), but also land accounting constraints described by (6). Algorithms by Meneghin et al. (1988) and Torres and Brodie (1990) may fail in application to the augmented adjacency matrix **A**', because they introduced their heuristic algorithms based on a two-dimensional map, which describes spatial relation among units. Utilizing the augmented adjacency matrix by Yoshimoto and Brodie (1990) may reduce additional adjacency constraints for the multiperiod scheduling problem.

Speaking of the computational effort, the more integer variables in the problem, the greater the computational time required to solve it by the B&B algorithm. Due to the limitations on the number of iterations, a solution could even be non-optimal. Because of this impracticability in integer optimization, heuristic algorithms to resolve the computational problem of providing a "good" solution have been developed. Since the future conditions of timber market or environmental conditions are uncertain, this "good" solution may be acceptable from a practical point of view. In what follows, a random search heuristic and a composite relaxation heuristic are reviewed.

The Random Search Technique

One category of heuristic technique is the random

search or Monte-Carlo search technique. Brooke (1958) discussed the usefulness of a random search technique compared to other nonlinear programming algorithms and concluded that when the model is of such complexity and/or size as to defy any attempt at an exact or approximate solution, it may be possible to apply the random method to yield an approximate solution backed up by a strong statement of confidence.

Bullard et al. (1985) applied this technique to optimize thinning regime in stand level analysis. At the forest level, Sessions and Sessions (1988), O'Hara et al. (1989) and Nelson et al. (1990) utilized a random search technique for a spatially constrained area-based forest planning problem.

A basic idea in common to these studies is to generate a set of feasible solutions in a random fashion, then select the solution with the highest objective value (for the maximization problem).

Sessions and Sessions' (1988) problem is a joint optimization problem of a road network and a spatially constrained harvest schedule with flow and adjacency constraints in the short term (over three periods). Their random search technique is the same as the original one (see Conley 1980), but is combined with a heuristic road network optimization (Sessions 1987). The procedure is such that it generates a feasible solution over three periods, then

applies a heuristic road network optimization to calculate the objective value for a final solution. Harvest flow level is given by the user, so that several trials with different flow levels are required to get an appropriate flow level. One of the reasons for limiting the problem to the short term might be that the more periods, the less probability that a solution is even feasible if each harvest unit is equally likely to be selected for harvest.

Due to the difficulty in generating a feasible solution over a long time horizon, O'Hara et al. (1989) applied a prebiasing search technique to favor harvest units considered more likely to yield good solutions for harvest scheduling problems with flow and adjacency constraints. Prebiasing is based on contribution of each harvest unit to the objective function, the number of adjacent units to each unit, or a combination of both, so that the probability of each unit to be selected for harvest is specified rather than randomly generated. They mentioned the use of binary search for determining the harvest flow level, but did not incorporate it into their model. In their examples, several trial-and-error procedures seem to be required to determine the timing and type of prebiasing methods to be applied. This poses the question of which prebiasing technique works best for any particular spatially constrained harvest scheduling problem.

Unlike the above two techniques by Sessions and

Sessions (1988) and O'Hara et al. (1989), Nelson et al. (1990) applied their random search technique to determine a tactical strategy in the short term, then combined it into FORPLAN (Johnson and Stuart 1987) for a long term sustainable solution. Harvest flow level in their problem is determined by initially solving the strata-based forest planning problem for the entire forest.

While the above two random search techniques determine harvest strategy at each period sequentially, a random search technique used in Nelson et al. (1990) applied a quasi-sequential look-ahead method to generate and improve a feasible solution toward a best solution. In other words, a quasi-sequential look-ahead method uses three best feasible solutions among a set of feasible solutions over all the periods (in their case three periods). Then only strategies at the first period of those three solutions are fixed. For the second period, three sets of feasible solutions over the rest of periods are generated, resulting from the previous three strategies. Then they select one best solution from each set. From these three best feasible solutions over the first two periods, harvest strategies for the following period are generated by satisfying feasibility over the rest of periods. This procedure continues until the last period. At the last period, among all feasible solutions, the one with the highest objective value is selected as an optimal solution. Note that all feasible

solutions at every period are derived from the three best feasible solutions over all the periods, so that at each iteration feasibility of a solution is guaranteed. When using this technique for a long-term problem, the solution time could increase dramatically due to the quasi-sequential look-ahead.

The Composite Relaxation Technique

Torres et al. (1990b) used a composite relaxation technique to solve the habitat dispersion problem, which is an area-based forest planning problem with wildlife habitat requirement constraints and harvest flow and adjacency constraints as well. Eliminating the wildlife habitat requirement constraints, the problem becomes the same as the proposed problem.

Their relaxation technique consists of; 1). dualizing complicated constraints (wildlife habitat requirement constraints and harvest flow constraints) into the objective function, 2). replacing the adjacency constraints by one surrogate constraint. The resultant problem, therefore, has a relaxed objective function and one surrogate constraint as well as land accounting constraints (harvest and reharvest decisions). The basic idea behind the composite relaxation technique is to transfer the problem into a simple knapsack problem, which has one objective function with only one

constraint.

Applying the original problem (IP) without land accounting constraints,

$$(IP) \qquad Z_{IP} = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} \} \tag{1}$$

subject to

$$\underline{V} \cdot \underline{X} = \underline{V0} \tag{2}$$

$$\underline{M} \cdot \underline{X} \leq \underline{M0} \tag{4}$$

$$\underline{X} \in \{0, 1\}$$

the composite relaxation problem is derived as follows. Dualizing constraints (2) with Lagrangean multipliers $\underline{\Pi}$, the first relaxed problem becomes a Lagrangean relaxation problem (LR) (Geoffrion 1974),

$$(LR) \qquad L(\underline{\Pi}) = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} - \underline{\Pi}^t \cdot \underline{V} \cdot \underline{X} \} \tag{7}$$

subject to

$$\underline{M} \cdot \underline{X} \leq \underline{M0} \tag{4}$$

$$\underline{X} \in \{0, 1\}$$

Changing multipliers $\underline{\Pi}$ provides several sets of $\underline{X}$ and the corresponding objective value $L(\cdot)$. The best set of multipliers $\underline{\Pi}^*$ should satisfy,

$$L(\underline{\Pi}^*) \leq L(\underline{\Pi}) \quad \text{for all } \underline{\Pi} \tag{8}$$

An alternative relaxation of the problem (IP) is formulated by a surrogate relaxation (Glover 1975) with surrogate multipliers $\underline{\sigma}$. Relaxing constraint (4), we have,

$$(S) \qquad S(\underline{\sigma}) = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} \} \tag{1}$$

subject to

$$\underline{V}\cdot\underline{X} = \underline{V0} \tag{2}$$

$$\underline{\sigma}^t\cdot\underline{M}\cdot\underline{X} \leq \underline{\sigma}^t\cdot\underline{M0} \tag{9}$$

$$\underline{X} \in \{0, 1\}$$

Like Lagrangean multipliers, the best set of surrogate multipliers $\underline{\sigma}^*$ is obtained by,

$$S(\underline{\sigma}^*) \leq S(\underline{\sigma}) \quad \text{for all } \underline{\sigma} \tag{10}$$

The Lagrangean relaxation reduces the number of constraints by bringing constraints into the objective function while the surrogate relaxation does so by transforming multiple constraints into a single constraint.

Combining the Lagrangean relaxation (LR) and the surrogate relaxation (S), a composite relaxation (Greenberg and Pierskalla 1970) is obtained by,

$$\text{(C)} \qquad C(\underline{\Pi},\underline{\sigma}) = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t\cdot\underline{X} - \underline{\Pi}^t\cdot\underline{V}\cdot\underline{X} \} \tag{7}$$

subject to

$$\underline{\sigma}^t\cdot\underline{M}\cdot\underline{X} \leq \underline{\sigma}^t\cdot\underline{M0} \tag{9}$$

$$\underline{X} \in \{0, 1\}$$

Similarly, changing a set of multipliers $\underline{\Pi}$ and $\underline{\sigma}$ the best set of multipliers $\underline{\Pi}^*$ and $\underline{\sigma}^*$ is obtained by,

$$C(\underline{\Pi}^*,\underline{\sigma}^*) \leq C(\underline{\Pi},\underline{\sigma}) \quad \text{for all } \underline{\Pi} \text{ and } \underline{\sigma} \tag{11}$$

The relationship among the above relaxation problems is described by Gavish and Pirkul (1985a),

$$L(\underline{\Pi}^*) \geq S(\underline{\sigma}^*) \geq C(\underline{\Pi}^*,\underline{\sigma}^*) \geq Z_{IP} \tag{12}$$

Torres et al.'s (1990b) formulation utilizes land accounting constraints with "no activity" as one decision,

$$\underline{L}\cdot\underline{X} = \underline{1} \tag{6}$$

separately to adjust the feasibility of the solution. As a result, their composite relaxation problem is actually formulated by,

$$C(\underline{\Pi},\underline{\sigma}) = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} - \underline{\Pi}^t \cdot \underline{V} \cdot \underline{X} \} \quad (7)$$

subject to

$$\underline{L} \cdot \underline{X} = \underline{1} \quad (6)$$

$$\underline{\sigma}^t \cdot \underline{M} \cdot \underline{X} \leq \underline{\sigma}^t \cdot \underline{M0} \quad (9)$$

$$\underline{X} \in \{0, 1\}$$

Their procedure starts finding surrogate multipliers $\underline{\sigma}$ first by solving,

$$S(\underline{\sigma}) = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} \} \quad (1)$$

subject to

$$\underline{\sigma}^t \cdot \underline{M} \cdot \underline{X} \leq \underline{\sigma}^t \cdot \underline{M0} \quad (9)$$

$$\underline{X} \in \{0, 1\}$$

The algorithm developed by Gavish and Pirkul (1985b) was used with modifications. Sorting adjacency constraints in terms of the objective values provided by maximizing $S(\underline{\sigma})$ with only one constraint, i.e., each adjacency constraint, the adjacency constraint with the lowest objective value is determined with the first surrogate constraint. Checking feasibility of a solution against the rest of the adjacency constraints, a surrogate constraint is updated until none of the adjacency constraints is violated by a solution.

Once surrogate multipliers are calculated, only Lagrangean multipliers are adjusted following the

subgradient method every time a solution is found iteratively. There is no iteration for searching for the best surrogate multipliers. Precise adjustment procedures for multipliers can be found in Torres et al. (1990a,b).

THE ALTERNATIVE TECHNIQUE BASED ON A RANDOM SEARCH TECHNIQUE

As can be observed, the shortcoming of a random search technique seems to be that it is difficult to generate even a feasible solution over long time horizons, and to provide an appropriate or optimal harvest flow level. In what follows, the solution strategy of our proposed technique, another modified random search technique, is introduced. The proposed technique not only solves a long term forest planning problem, but also provides an appropriate harvest flow level simultaneously.

Projection Alternative TecHnique (PATH) in Harvest Scheduling Problems

The PATH (Projection Alternative TecHnique) algorithm was originally derived for the stand level optimization problem (Paredes and Brodie 1987, and Yoshimoto et al. 1988). By eliminating projections from inferior alternatives over the entire projection period and redefining the objective function, the computational task was vastly reduced. The basic concept behind the PATH algorithm can be interpreted in such a way that the problem can be partitioned into several subproblems stage by stage or period by period with the iterative process over the stage or period. One problem remains as to how to redefine

the objective function of a subproblem in order to achieve an optimal solution.

Applying PATH to the harvest scheduling problem, the solution technique should consist of two phases. One is global optimization, and the other is regional optimization. By applying the one-stage look-ahead technique from PATH, the problem is partitioned into a subproblem at each period where regional optimization generates a feasible solution for each period. The subproblem at the regional optimization phase is specified as follows.

At the regional optimization phase, the subproblem could be formulated by partitioning the original problem (1) directly into,

$$Z_{IP}(i) = \underset{\{\underline{X}_i\}}{\text{maximize}} \{ \underline{C}_i^t \cdot \underline{X}_i \} \tag{13}$$

subject to

$$\underline{M}_i \cdot \underline{X}_i \leq \underline{M0}_i \tag{14}$$

$$\underline{V}_i \cdot \underline{X}_i = \underline{V0}_i \tag{15}$$

$$\underline{X}_i \in \{0,1\}$$

where $\underline{X}_i$ is a vector representing only a solution for the i-th period. $\underline{C}_i$ is a coefficient vector of $\underline{C}$, $\underline{M}_i$ is a sub-matrix of $\underline{M}$, $\underline{M0}_i$ is the corresponding vector of $\underline{M0}$, and $\underline{V}_i$ is a sub-matrix of $\underline{V}$, for $\underline{X}_i$, respectively at the i-th period. Since land accounting constraints can not be specified period by period, they should be handled implicitly. The random search technique could then be applied to the above subproblem.

Unlike the stand level analysis (Yoshimoto et al. 1990), due to the harvest flow constraints and adjacency constraints, the subproblem cannot be treated independently. In other words, a solution of the above problem is most unlikely to be feasible for the following period. This is because of the fact that if "good" harvest units with high present net worth are always selected for harvest first, then the rest of units would not be able to sustain the even-flow level (Nelson 1988). A modification of problem specification can be implemented by changing the objective function and feasibility conditions of the subproblem.

Since even-flow constraints are most likely to be violated, these concerns can be respecified as minimization of flow deviation. The resultant problem becomes a multicriteria problem which maximizes the present net worth and minimizes the deviation of harvest flows over periods. In the proposed technique, minimization of the harvest flow deviation becomes the objective of the subproblem (SP). In addition, feasibility is guaranteed in the following period as well as the current period. These two modifications of the subproblem will guide the random search technique to generate a "good" feasible solution. Note that the feasibility condition could be further expanded, however as the number of periods for feasibility increases, the computational time increases exponentially, and the random search technique becomes complicated and inconvenient in

application. Thus a feasibility condition for only two sequential periods is used.

The subproblem (SP) at the i-th period is formulated by,

$$(SP) \qquad Z_{IP}(i) = \underset{\{\underline{X}_i\}}{\text{minimize}} \; | \; \underline{V}_i \cdot \underline{X}_i - V1 \; | \qquad (16)$$

subject to

$$\underline{M}_i \cdot \underline{X}_i \leq \underline{M0}_i \qquad (14)$$

$$\underline{M}_{i+1} \cdot \underline{X}_{i+1} \leq \underline{M0}_{i+1} \qquad (17)$$

$$(1-P) \cdot V0 \leq \underline{V}_i \cdot \underline{X}_i \leq (1+P) \cdot V0 \qquad (18)$$

$$(1-P) \cdot V0 \leq \underline{V}_{i+1} \cdot \underline{X}_{i+1} \leq (1+P) \cdot V0 \qquad (19)$$

$$\underline{X}_i, \; \underline{X}_{i+1} \in \{0,1\}$$

where P is an acceptable percentage of harvest flow fluctuation. $|\cdot|$ denotes the absolute value, and V1 is set as $(1-P) \cdot V0$, the minimum level of harvest flow in our model. As can be seen, a feasible solution at each period is not only feasible in the current period but also feasible at the following period in a sense that it can generate at least one feasible solution in the following period.

Minimizing Deviation of Harvest Flow

The random search technique called ROHO (Random Ordering Heuristic Optimization) is applied for the subproblem to generate a set of feasible solutions at each period. ROHO works as follows. Assigning a random value to harvest units, a descending ordered sequence is obtained

based on a random value. The ordered sequence is used to select units for harvest. In other words, at first, the first ordered unit is selected for harvest. Eliminating the adjacent units to the selected unit, a unit with the highest random value among the rest of the candidates is selected as the second unit. At each selection, units adjacent to the units previously selected are eliminated. Since the objective of the subproblem is to minimize a deviation of harvest flow, once the total harvestable volume becomes greater than a given harvest level, V1, selection of units ceases. Since the order, by which one harvest unit is selected affects selection of the rest of the units, a descending order is changed by replacing the first ordered unit into the last one, the second ordered into the first, and so on. After obtaining a new ordered sequence, another trial to generate a feasible solution is implemented. This changing procedure continues until the first ordered sequence appears again. As a result, at least once, each unit dominates the first selection for a solution with a set of given random numbers. The following is a changing procedure of the ordered sequence.

1st sequence: X(1),X(2),X(3),,,,X(n)

2nd sequence: X(2),X(3),X(4),,,,X(1)

3rd sequence: X(3),X(4),X(5),,,X(1),X(2)

.
.
.

n-th sequence:X(n),X(1),X(2),,,X(n-1)

where X(i) is the i-th ordered harvest unit. Given a sequence, after a feasible solution is generated for the current period, the same trial is implemented to see if this feasible solution can provide a feasible solution for the following period. If there is no feasible solution for the following period, a solution for the current period becomes infeasible. After completion of this iteration, the ordered sequence can also be arranged by assigning another set of random numbers to units.

By changing the ordered sequence, a set of feasible solutions for the current period and the following period is obtained. Among them a solution with the closest current harvest flow to a given flow level, V1, i.e., minimum deviation of harvest flow, is chosen as the best solution at the current period and the remaining periods. A flowchart of the ROHO algorithm is presented in Figure 3.1.

In comparison to the other random search technique, ROHO sets up the ordered sequence randomly, but not the selection of a harvest unit. The stochastic nature of the random search technique does not influence the selection of harvest units given an ordered sequence. Although ROHO and the other random search techniques cease selection of harvest units at each period when the harvestable volume exceeds the lower bound of the flow constraints, the solution from ROHO has the "minimum" deviation from the lower bound among a set of feasible solutions. However, the

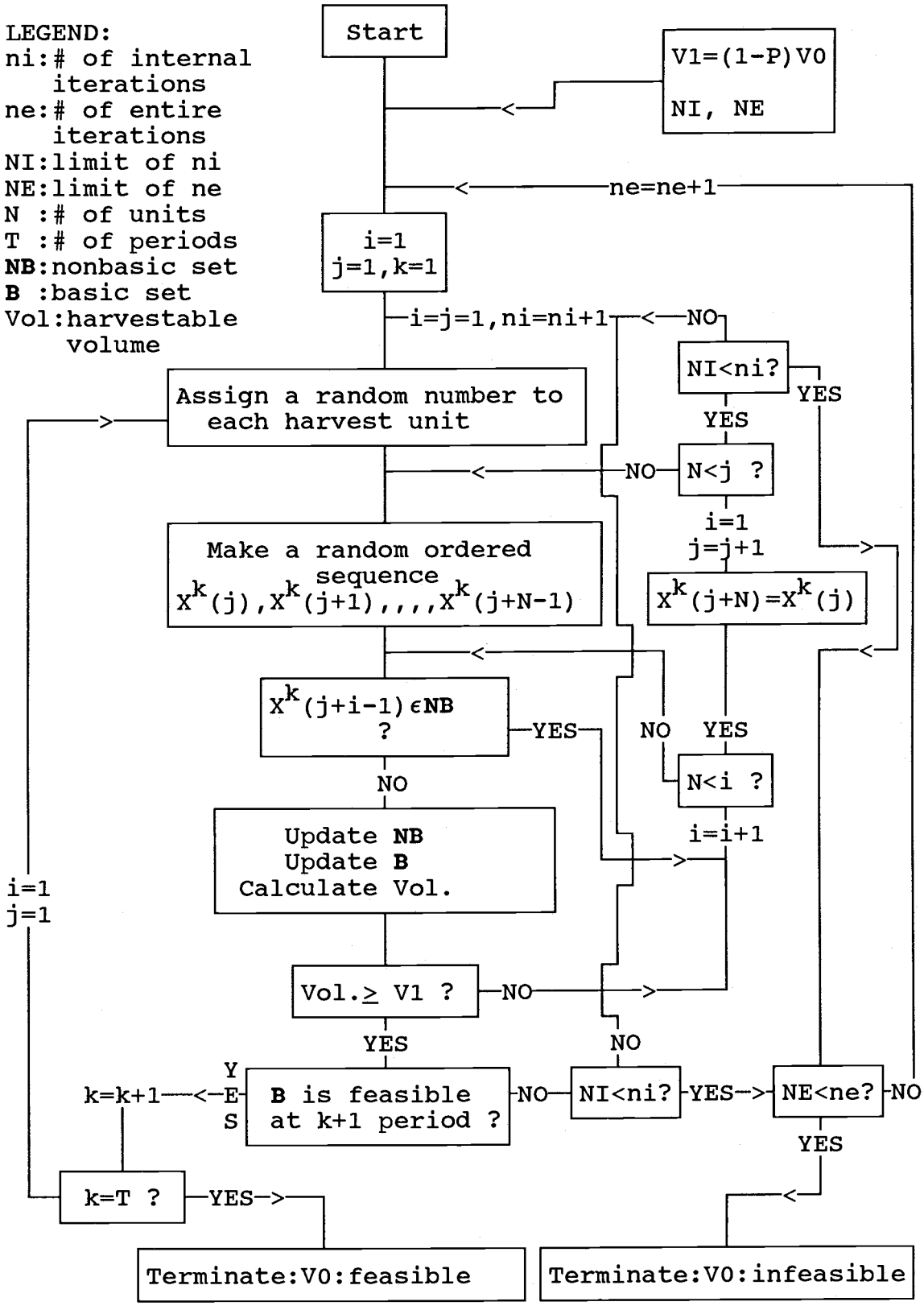


Figure 3.1. The structure of ROHO

other techniques do not have any particular solution characteristics other than feasibility.

The Binary Search Method For The Harvest Flow Level

To search for a "good" solution over the time horizon, the proposed technique combines the ROHO algorithm and the PATH algorithm, called ROHO-PATH (Random Ordering Heuristic Optimization with Projection Alternative TecHnique) for a given harvest flow level, V0.

By changing the harvest flow level, V0, using the binary search method at what we will call the global optimization phase in the ROHO-PATH algorithm, a set of feasible solutions with different objective value and with "minimum" deviation of harvest flow over the entire period, is created.

The binary search method to determine an appropriate or optimal harvest flow level, V0 with some modification is described as follows. As an initial step, an initial harvest flow level, $V0^0$ is to be specified. An acceptable percentage, P, of harvest flow fluctuation for a given harvest level is also to be specified. As long as harvest flow level at each period lies within $\pm$ P % from a given harvest level, $V0^k$, a solution is regarded as a feasible solution. Once a feasible solution is generated over all periods, a new harvest flow level is determined by the

following incremental sequence,

$$(1 - P) \cdot V0^{k+1} = (1 + P) \cdot V0^{k}$$

or

$$V0^{k+1} = \frac{(1+P)}{(1-P)} V0^{k} \qquad \text{for } k \geq 0 \qquad (20)$$

That is, a minimum harvest flow level for a new iteration, $V0^{k+1}$ is set equal to a maximum harvest flow level for a previous iteration, $V0^{k}$. If there is no feasible solution after a given number of trials, harvest flow level is reduced to a previous level, and then a new flow level is specified by increasing this level by one percent until the total number of sequential failures reaches a given number. Since the problem is discrete leading to an irregular response surface, the method of sequential increase in V0 by a constant rate works better than the original binary search method, which works well for a smooth unimodal response surface.

Maximizing Present Net Worth

For a given even-flow level, V0, the iterative procedure of minimizing deviation of harvest flow at each period creates one feasible solution for all the periods if the even-flow level is feasible. Applying a modified binary search method for optimizing an even-flow level with "minimum" deviation of harvest flow, a set of feasible solutions for all periods with different flow level is

obtained. Among them, the best solution is to be selected as a final solution based on a certain criterion.

Torres et al. (1990a) provide a criterion to distinguish one primal infeasible solution from another infeasible solution to the original IP problem with even-flow constraints. They observed that the primal solution associated with a Lagrangean problem with minimum value, which in addition has the smallest sum of absolute infeasibility, is a better primal solution than a solution with larger objective function value and larger sum of absolute infeasibility. As a result, the sum of absolute infeasibilities was used as a criterion in their problem.

In our problem, however, the minimum sum of absolute infeasibilities is not always preferable. For instance, suppose the problem is a four period problem, and the first primal infeasible solution has the following harvest flow pattern,

530 mbf at 1st period
500 mbf at 2nd period
500 mbf at 3rd period
500 mbf at 4th period

and an even-flow level is 500 mbf. The sum of absolute infeasibilities among periods is calculated as 30 mbf. In addition, the sum of absolute infeasibilities from the 500 mbf even-flow level is also 30 mbf. Further suppose the second infeasible solution has the following harvest flow

pattern,

515 mbf at 1st period

505 mbf at 2nd period

515 mbf at 3rd period

505 mbf at 4th period

and the even-flow level is the same as the other solution. The sum of absolute infeasibility among periods is the same as the first solution, 30 mbf, but there is a 40 mbf deviation from the 500 mbf even-flow level. Although both solutions have the same sum of absolute infeasibilities among periods and the first solution has less infeasibility from the 500 mbf even-flow level, the second solution is preferable because of less fluctuation of harvest flow. In other words, infeasibility is spread over all the periods.

In considering spread of infeasibility over periods, one of the alternatives for a criterion can be percentage of harvest flow FLUCtuation, FLUC (%), calculated by,

$$\text{FLUC}(\%) = \frac{\text{Flow.max} - \text{Flow.min}}{\text{Flow.max} + \text{Flow.min}} * 100(\%) \qquad (21)$$

where Flow.max is a maximum harvest flow during periods and Flow.min is a minimum flow. Actually, FLUC is a solution of the following system of equations for FLUC and flow level V,

$$(1 - \text{FLUC}/100)\cdot V = \text{Flow.min} \qquad (22)$$

$$(1 + \text{FLUC}/100)\cdot V = \text{Flow.max} \qquad (23)$$

Thus FLUC reflects the fact that all harvest flows lie within $\pm$ FLUC (%) fluctuation from V, and FLUC is the minimum fluctuation level. The smaller FLUC, the less

fluctuation in harvest flow. For the above example, the first solution has FLUC(1) = 2.9 % and the second has FLUC(2) = .9 %. Thus based on the minimum fluctuation criterion, the second primal infeasible solution is determined as a better solution than the first one.

When ROHO-PATH generates a feasible solution, the acceptable harvest fluctuation level, P, in the subproblem (SP) is first specified. In addition, ROHO-PATH uses further restricted fluctuation level, P' (< P), dividing a set of feasible solutions into two groups,

$$G1 = \{\underline{X} : P' < \text{percent of fluctuation} \leq P\}$$

$$G2 = \{\underline{X} : 0 \leq \text{percent of fluctuation} \leq P'\}$$

Using the minimum fluctuation criterion, the second group is better than the other. The solution with the highest objective value in G2 is regarded as a final solution. If there is no such solution in G2, a solution with the highest objective value in G1 is regarded as a final solution. The minimum fluctuation criterion is only used to divide a set of feasible solutions by ROHO-PATH into these two groups.

Due to the discrete nature of the problem, there is a case where ROHO-PATH with a small P value often fails to generate a feasible solution to the subproblem, leading to the failure of a search for the best solution. With two different levels of flow fluctuations, we can not only classify a set of feasible solutions but also let ROHO-PATH continue to search a solution with the maximum allowable

level of fluctuation, P, and obtain an optimal solution with the preferable level of fluctuation, P'. That is, P is to generate a set of feasible solutions, and P' is to select the final solution by maximizing the objective function. The structure of the ROHO-PATH algorithm is described in Figure 3.2.

Since the original problem

$$\text{(IP)} \qquad Z_{IP} = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} \} \tag{1}$$

subject to

$$\underline{V} \cdot \underline{X} = \underline{V0} \tag{2}$$

$$\underline{M} \cdot \underline{X} \leq \underline{M0} \tag{4}$$

$$\underline{L} \cdot \underline{X} \leq \underline{1} \tag{6}$$

$$\underline{X} \in \{0, 1\}$$

is transformed into

$$Z_{IP} = \underset{\{\underline{X}\}}{\text{maximize}} \{ \underline{C}^t \cdot \underline{X} \} = \underset{\{\underline{X}_i\}}{\text{maximize}} \{ \sum_{i=1}^{T} \underline{C}_i^t \cdot \underline{X}_i \} \tag{24}$$

where $\underline{X}_i$ is a solution of

$$\text{(SP)} \qquad Z_{IP}(i) = \underset{\{\underline{X}_i\}}{\text{minimize}} \left| \underline{V}_i \cdot \underline{X}_i - V1 \right| \tag{16}$$

subject to

$$\underline{M}_i \cdot \underline{X}_i \leq \underline{M0}_i \tag{14}$$

$$\underline{M}_{i+1} \cdot \underline{X}_{i+1} \leq \underline{M0}_{i+1} \tag{17}$$

$$(1-P) \cdot V0 \leq \underline{V}_i \cdot \underline{X}_i \leq (1+P) \cdot V0 \tag{18}$$

$$(1-P) \cdot V0 \leq \underline{V}_{i+1} \cdot \underline{X}_{i+1} \leq (1+P) \cdot V0 \tag{19}$$

$$\underline{X}_i, \ \underline{X}_{i+1} \in \{0,1\}$$

the final solution of the proposed problem is not exactly

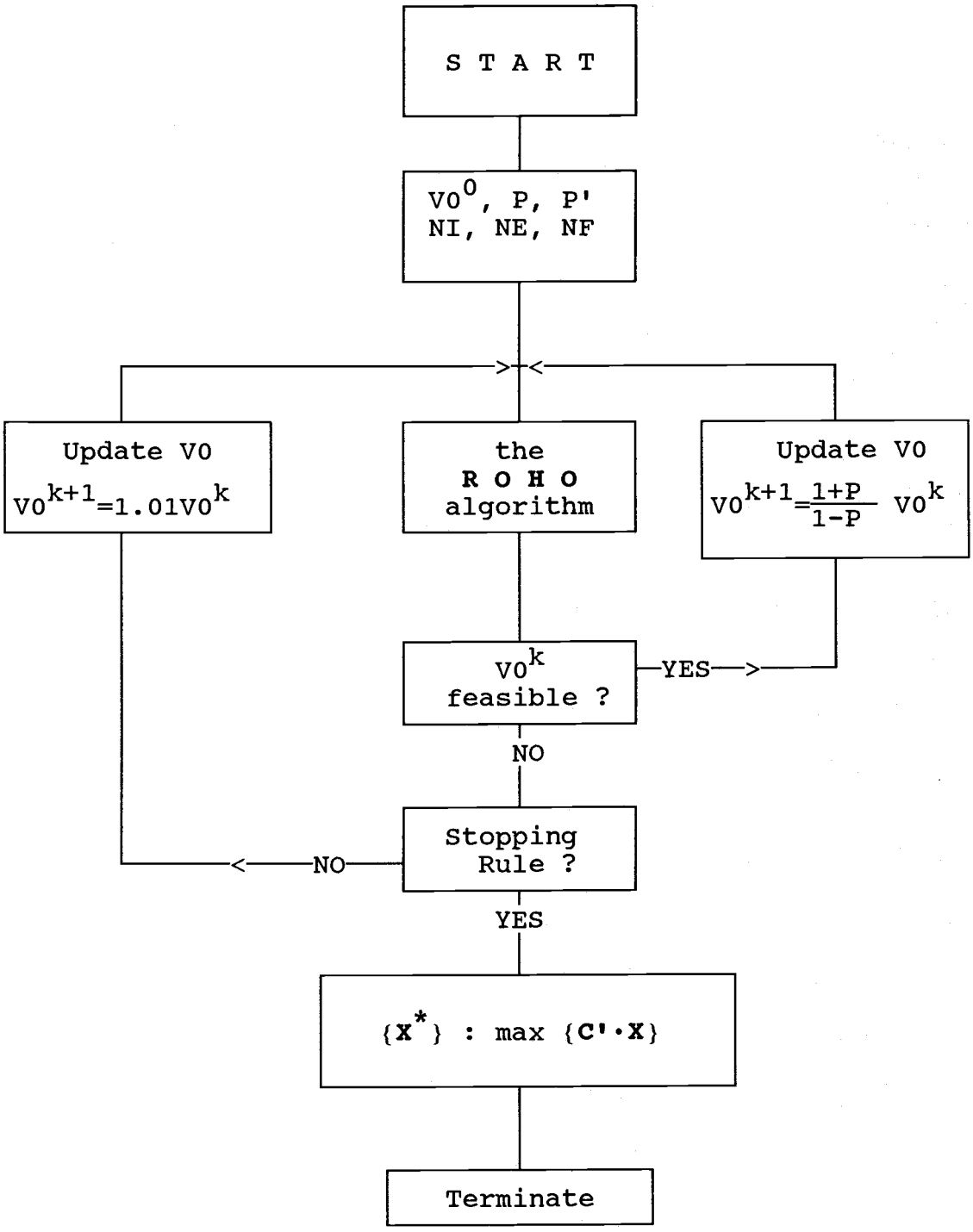


Figure 3.2. The structure of the ROHO-PATH algorithm

the same as the solution of the original one. However, with small violations of harvest flow constraints, a set of solutions $\{\underline{X}_1, \underline{X}_2, , , , , \underline{X}_T\}$ generated by the proposed technique, should be at least near-feasible and "good" for the original problem.

MODEL DEVELOPMENT

Applying the ROHO-PATH algorithm, the spatially constrained area-based harvest scheduling model called SSMART (Scheduling System of Management Alternatives foR Timber-harvest) was developed to handle 500 maximum harvest units up to 20 periods. SSMART was written in FORTRAN 77 and compiled by RM/FORTRAN Compiler Version 2.43. A single-tree/distance-independent growth model, Stand Projection System (SPS) (Arney 1985) was incorporated into SSMART as an option to generate volume and corresponding value of stands. A random number generator by Schrage (1979) was used for the ROHO part of SSMART.

When using SPS, value data for a stand is calculated with price equation as a function of diameter-breast-height (dbh),

$$\text{Price} = f(\text{dbh})$$

Price can be specified on the basis of either thousand cubic feet or thousand merchantable board feet. Regeneration cost and interest rate are specified by the user as well as the number of periods, the minimum rotation periods, and the number of periods for adjacency lag, during which adjacent harvest units cannot be harvested.

SSMART has three main parameters to modify the efficiency of the iterative process. The first parameter is the maximum number of internal iteration, NI, which

determines the number of sets of random numbers assigned to harvest units. Utilizing ROHO at each period, NI ordered sequences are obtained. NI times n (number of units) is the upper bound of the number of feasible solutions at each period. Among those feasible solutions, the one with the closest harvestable volume to a lower bound of flow level is selected at each period.

The second parameter is the maximum number of entire iterations over the time horizon, NE. In order to decide if a given even-flow level, V0, is feasible, SSMART continues the iterative process until the total number of failures to search for a feasible solution reaches NE.

The third parameter is the maximum number of sequential failures, NF, for a flow level V0. After the total number of failures to search for a solution reaches NE, SSMART keeps increasing V0 by one percent for another trial, but with only one trial over the time horizon, i.e., NE = 1. Once the number of sequential failures reaches NF, SSMART ceases the process, then selects the best solution with the highest objective value among a set of feasible solutions which satisfy P' of flow fluctuation. If any solution is found before the number of sequential failures becomes NF, NE is reset as before, then SSMART continues the iterative process for another solution.

An initial harvest flow level is specified by dividing the total volume at the first period by the number of

periods, provided that the number of periods is more than two. Otherwise, the total volume is divided by three for an initial harvest flow level, V0. This can be changed by the user depending on the problem.

MODEL EXPERIMENTATION

To evaluate the performance of the ROHO-PATH algorithm by SSMART as an alternative technique for seeking the optimal solution of forest planning problems, two forest structures are adopted from Nelson et al. (1990) and Barker (1989).

The first forest is located on the southern coast of British Columbia. This forest consists of 109 stands grouped into 62 analysis areas. Among them, 45 harvestable units are used for spatially constrained area-based plans. Spatial relationship among units is depicted in Figure 3.3. As can be seen, 45 units are divided into two groups, resulting in a reduction of the number of adjacency relationships. Species of all trees are assumed to be Douglas-fir (*Pseudotsuga menziesii* (Mirb) Franco). Total area of the 45 units is 4190 acres. Stand age ranges from 60 years to 200 years. Age distribution at the current period is given in Figure 3.4-a. Each stand structure was derived from a stand with site index 135 feet at breast height age basis by using the SPS growth model. Table 3.1 shows this base stand input file for SPS.

The second forest is the Green River Subbasin on the Alsea Ranger District, Siuslaw National Forest in Oregon. Forest structure is depicted in Figure 3.5. The total area is 6382 acres. This forest includes five private blocks.

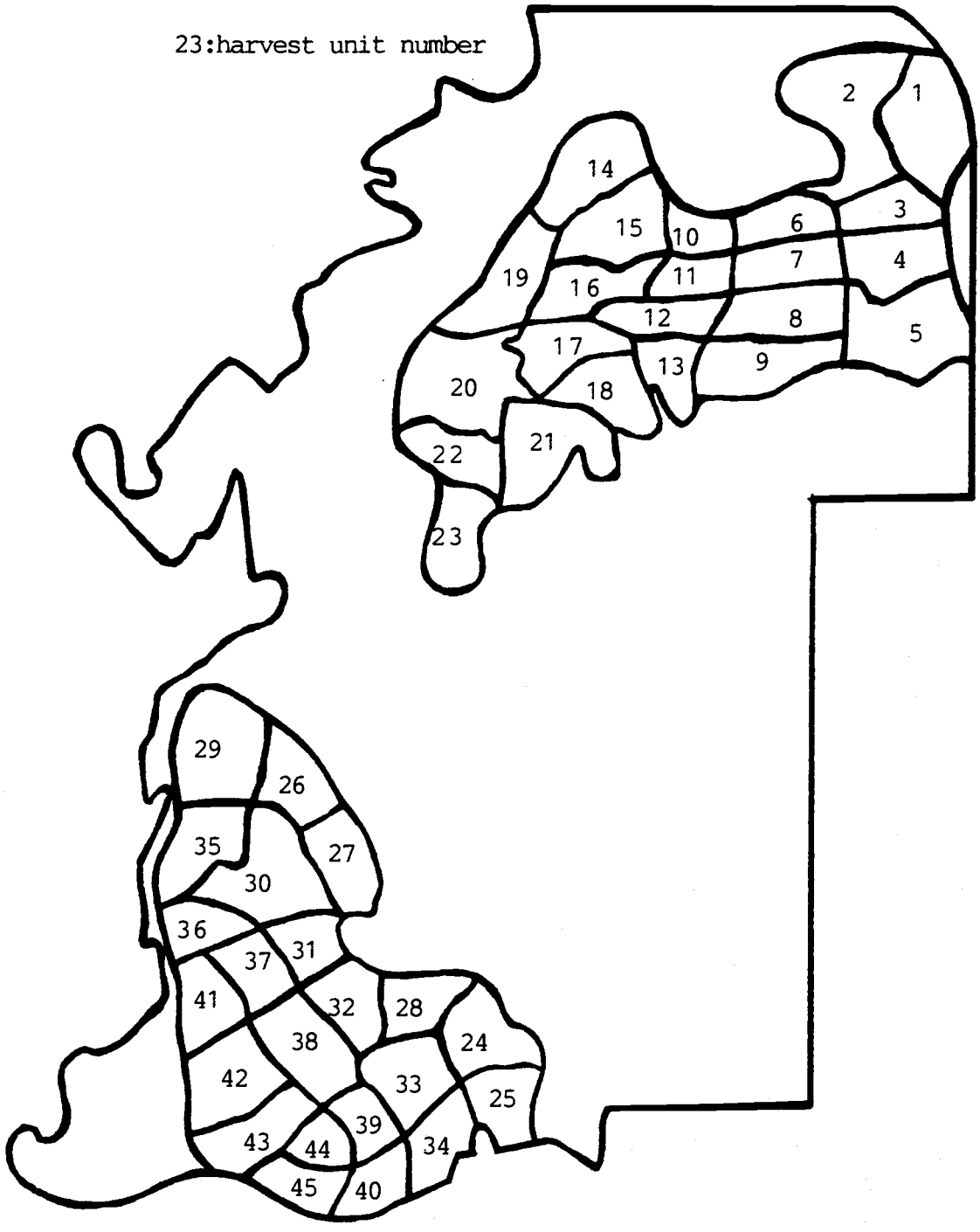
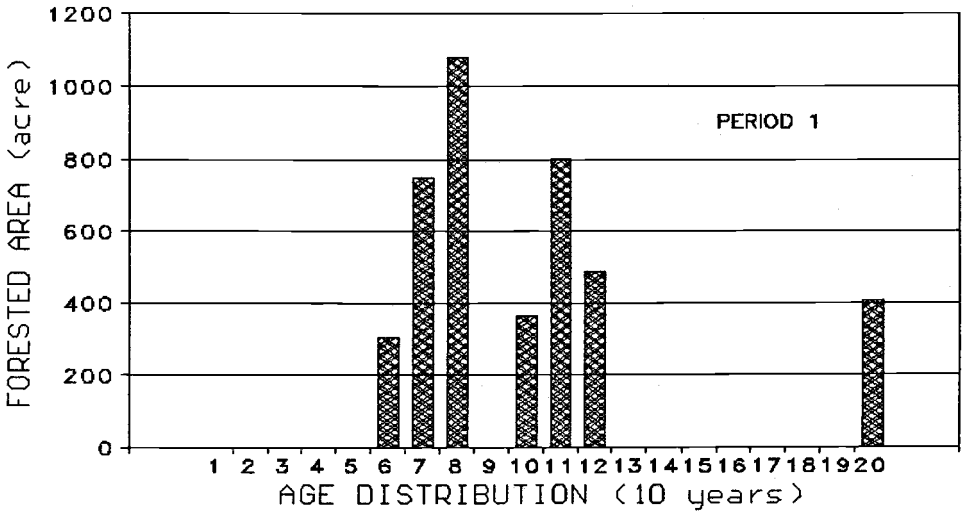


Figure 3.3. The first example forest from Nelson et al.(1990)

a)



b)

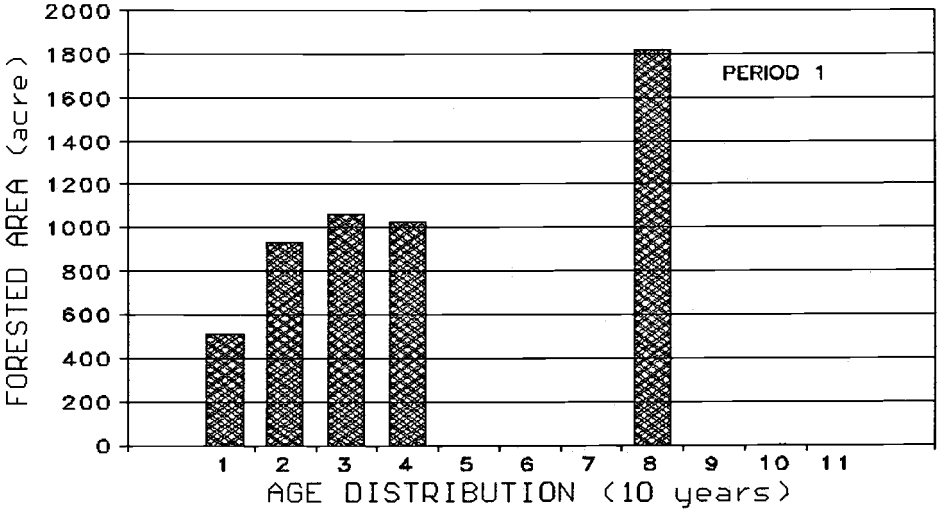


Figure 3.4. The stand age distribution
a) The first example forest (Nelson et al. 1990)
b) The second example forest (Barker 1989)

Table 3.1. Input data file for SPS (see Arney 1985)

```
             SP SITE  AGE REGION
STAND        DF  135   10  PNW
MERCH       1.0 16.4  4.0  6.0
NAME      input data file for SPS

             SP  DBH  TOP TPA   AGE SDEV   NAT
TABLE    1   DF  1.0 30.0 100.   10  .25    1
TABLE    2   DF  2.0 32.0 150.   10  .25    1
TABLE    3   DF  3.0 33.0  50.   10  .25    1

CLUMP       .90
REPORTS      10   20   30   40   50   60   80
```

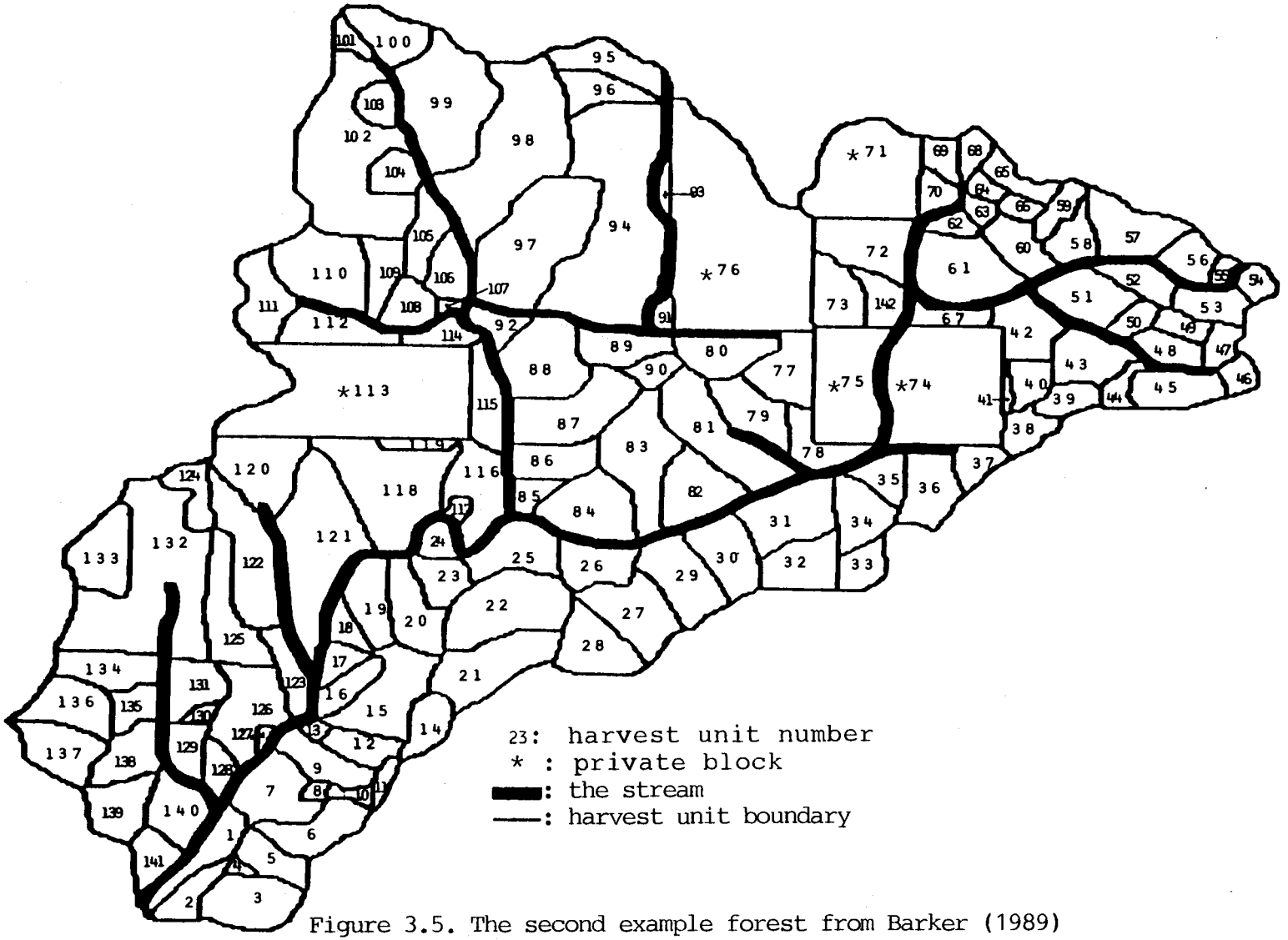


Figure 3.5. The second example forest from Barker (1989)

Species of all trees are also assumed to be Douglas-fir. The number of harvestable units is 137 without the private blocks. There are two age groups of stands. One is mature stands above age 80, and the other is young stands with age distribution ranging from 10 to 40 years (Figure 3.4-b). Stand volume also is created based on the stand in Table 3.1 by SPS.

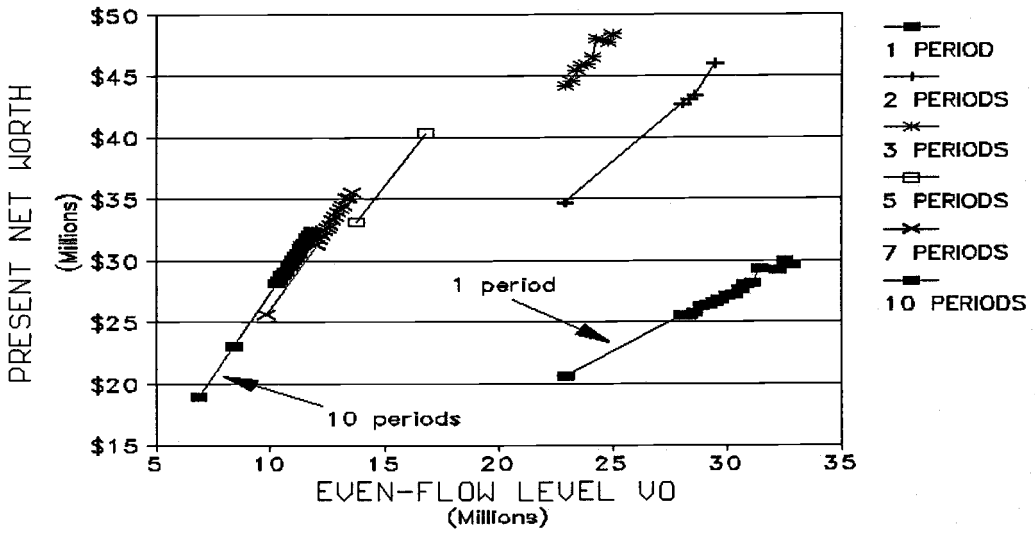
Price per cubic foot was unity for all diameter classes. No price premium by dbh was assumed. A four percent interest rate was used. No cost was imposed. Six different time horizons were used, i.e., one, two, three, five, seven and ten decades. The minimum rotation age was set as 60 years (6 periods). Thus, there was no regeneration harvest for the 1-, 2-, 3-, and 5-period problems. The number of periods for which adjacent units must lag between harvests was set as one.

As a criterion for a feasible solution in SSMART, 10 % flow fluctuation was used as the first acceptable level of fluctuation, P, and 5 % was used for the second preferable level, P'. That is, a set of feasible solutions should satisfy $\pm$ 10 % fluctuation over the time horizon, and a restricted set should satisfy $\pm$ 5 %. The final solution is selected from a restricted set with the highest present net worth. Three different NI's, 1,2, and 3 were used for the stopping rule of the ROHO algorithm at each period. NE was set as 10, and NF was 5.

The forest from Nelson et al. (1990) was used first. Figure 3.6 shows the relationship between the objective value and even-flow level, V0. In Figure 3.6-a, NI =1, while NI = 3 in Figure 3.6-b. As can be seen in Figure 3.6, all changing patterns for NI = 1, and NI = 3, are the same. The initial large change in the objective value for the 1-, 2-, 5-, 7- and 10-period problems was caused by the incremental change in V0 from equation (20). The three-period problem did not have this large initial change in V0, while the 10-period problem had two large changes. For the 5-period problem, no feasible solutions were found after the large change. Changing the initial even-flow level, V0, might change the number of big jumps on points, which also affect the computational time. Judging from the final objective value, there is not much difference between solutions by NI = 1 and NI = 3.

As for the quality of feasible solutions in terms of flow fluctuation, however, there is some difference between them. Figure 3.7 shows the change in flow fluctuation over the number of trials. Note that the more the number of trials, the larger the even-flow level, V0. The upper graph, Figure 3.7-a, shows high deviation on flow fluctuation of feasible solutions, while the lower graph does not. The average flow fluctuation of feasible solutions for NI = 1 is 1.3 (%) with 1.00 (%) standard deviation. For NI = 3, the average is 0.6 (%) with 0.5 (%)

a)



b)

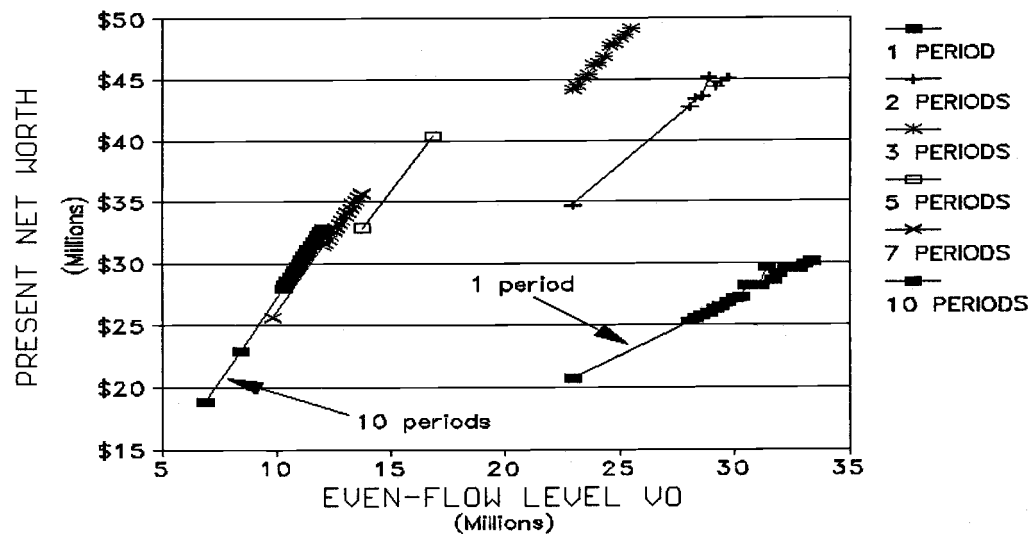
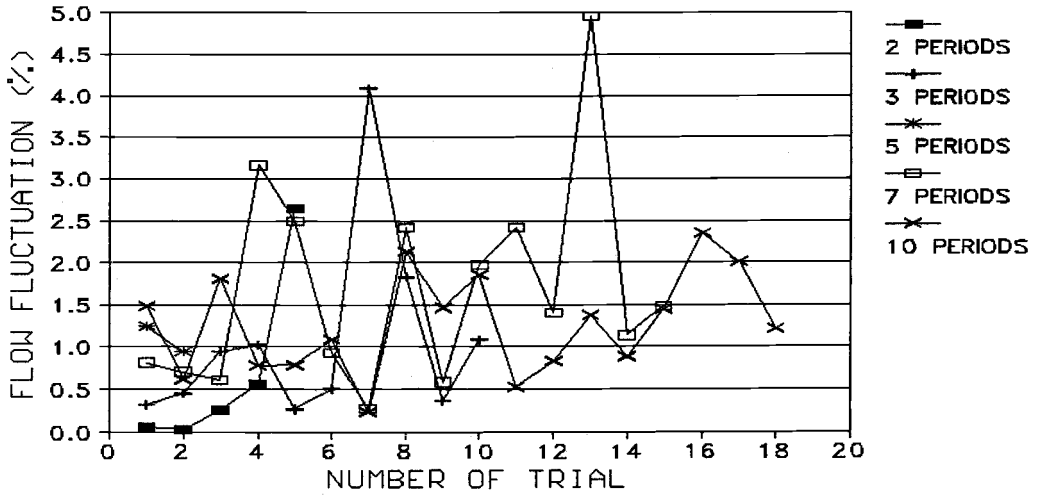


Figure 3.6. The relation between the objective value and even-flow level V0 for the first example forest by SSMART
a) NI = 1
b) NI = 3

a)



b)

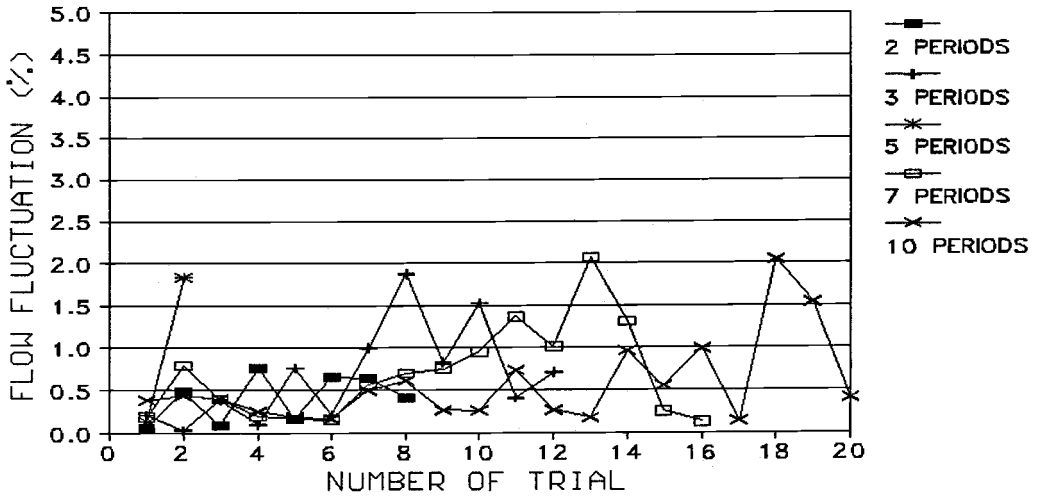


Figure 3.7. Changing patterns of flow fluctuation for the first example forest by SSMART
a) NI = 1
b) NI = 3

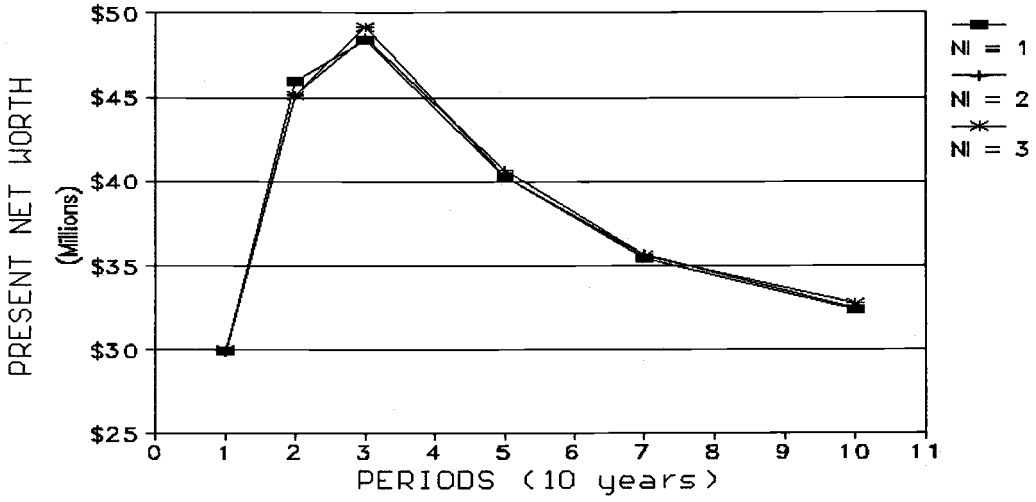
standard deviation. From this result, it can be said that the larger NI, the less the flow fluctuation. This may be due to the fact that for larger NI, more feasible solutions may be generated, resulting in a high probability of having a solution with lesser flow fluctuation.

Figure 3.8 shows three sets of final solutions for 6 different period problems. Figure 3.8-a is in terms of the objective value and Figure 3.8-b is in terms of the calculational time (seconds). Difference in the objective value among three different NI's is small relative to the absolute value of the objective value. The largest difference was 1.89 % and the smallest was 0.47 %. On the other hand, the calculation time is not so. The larger NI, the more the iterations. However, doubling NI does not mean twice the calculation time.

Based on the objective value, NI = 3 provided the highest value for the 1-, 3-, and 10-period problems, while NI = 2 does so for the 5- and 7- period problems, and NI = 1 does for the 2-period problem. All final solutions are given in Table 3.2.

Using a forest from Barker (1989), the second experiment was completed. Figure 3.9 shows the same relationship as Figure 3.6. For different NI, both graphs in Figure 3.9 were almost the same. In comparison to Figure 3.6, because of the availability of harvestable units in Barker's (1989) forest, final even-flow level V0 for the 5-,

a)



b)

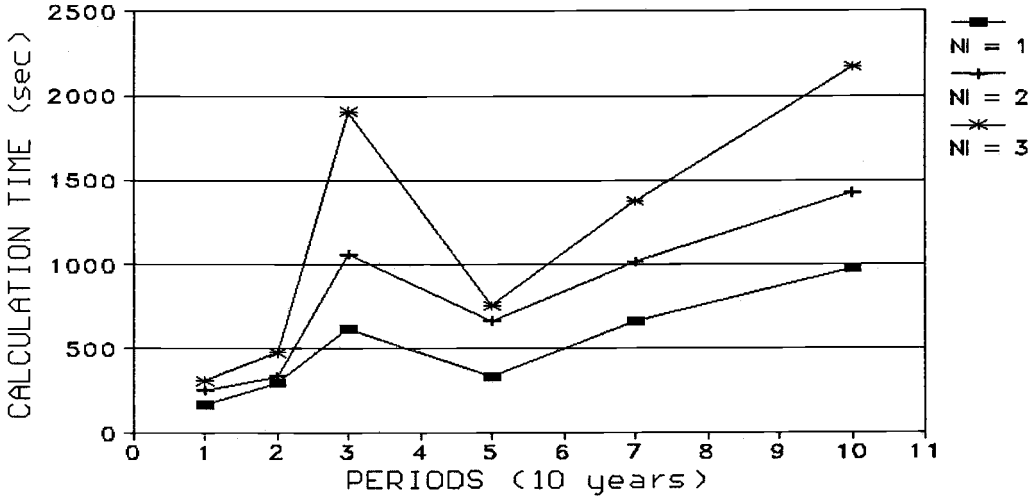


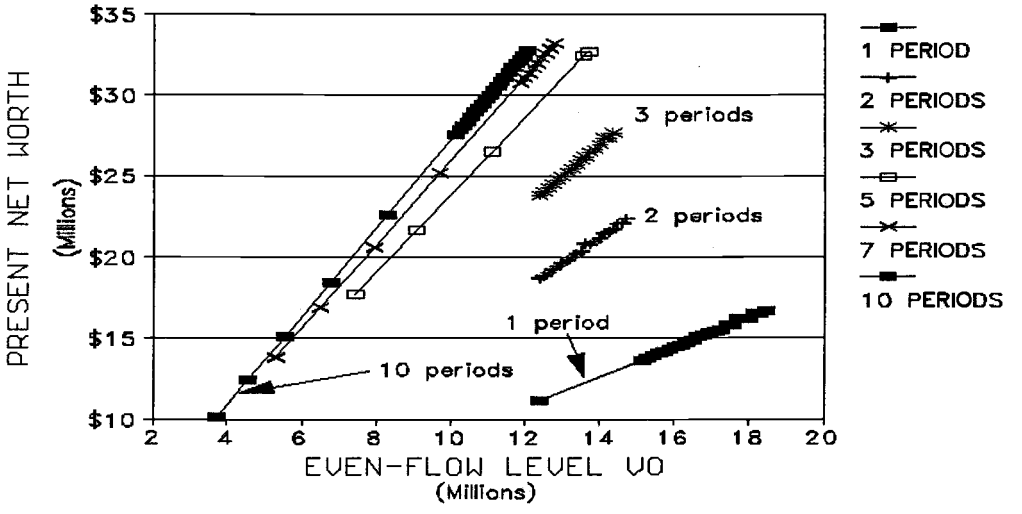
Figure 3.8. Comparison of solutions by SSMART for the first example forest
a) In terms of the objective value
b) In terms of the computational time

Table 3.2. The final solutions for the first example forest by SSMART

1) Period	2) # of Integer Vars.		1 Internal Iteration	2 Internal Iterations	3 Internal Iterations	Best Solution
1	45	Objective Value	29964798	29908054	30048438	30048438
		Fluctuation (%)	N/A	N/A	N/A	N/A
		Time (second)	164	252	312	312
2	90	Objective Value	45985504	45117420	45117420	45985504
		Fluctuation (%)	2.64	0.16	0.16	2.64
		Time (second)	299	338	480	299
3	135	Objective Value	48398436	48512456	49174844	49174844
		Fluctuation (%)	1.09	0.12	0.71	0.71
		Time (second)	620	1061	1909	1909
5	225	Objective Value	40299364	40659432	40319384	40659432
		Fluctuation (%)	0.95	0.59	1.84	0.59
		Time (second)	338	670	755	670
7	360	Objective Value	35485656	35680836	35615752	35680836
		Fluctuation (%)	1.49	1.94	0.13	1.94
		Time (second)	663	1016	1380	1016
10	900	Objective Value	32423532	32520914	32767306	32767306
		Fluctuation (%)	1.22	0.41	0.41	0.41
		Time (second)	980	1425	2171	2171

1): the number of periods
2): the number of integer variables

a)



b)

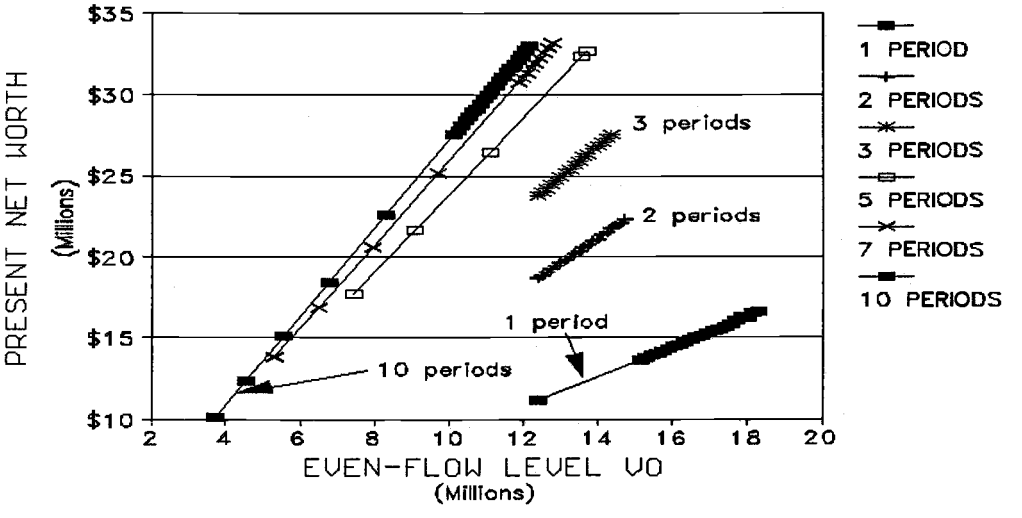


Figure 3.9. The relation between the objective value and even-flow level V0 for the second example forest by SSMART
a) NI = 1
b) NI = 3

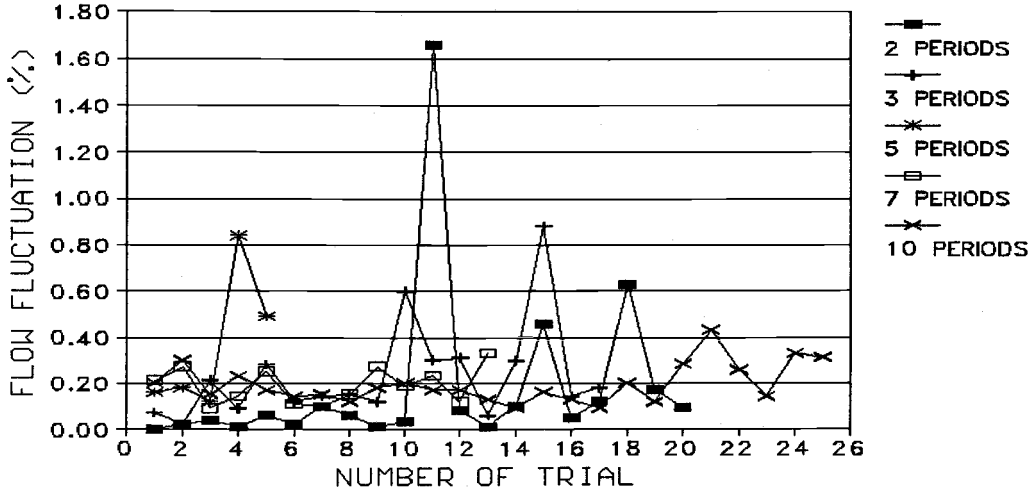
7-, and 10-period problems were closer to the 1-, 2-, and 3-period problems than those in Figure 3.6.

From Figure 3.10, it is also observable that the larger NI, the less the flow fluctuation with lower standard deviation. For NI = 1, the average is 0.21 (%) with 0.23 (%) standard deviation, while the average is 0.10 (%) with standard deviation 0.11 (%) for NI = 3. Again, because of the greater availability of harvestable units, these figures are much less than those from Nelson et al.'s (1990) forest.

Three sets of final solutions are depicted in Figure 3.11. Figure 3.11-a is based on the objective value and Figure 3.11-b is based on the calculation time (seconds). NI = 1 provided the highest objective value for the 1-, 3-, 5- and 7-period problems, while NI = 2 does for the 2- and 10-period problems. There are no superior solutions by NI = 3 (Table 3.3). The largest difference of the objective value is 2.24 % and the smallest is 0.06 %.

Figure 3.12 shows the dynamics of stand age distribution for Nelson et al.'s (1990) forest over 10 periods. At period 1, the forest does not have stands younger than 50 years, at 90 years, and between 130 and 190 years. The oldest stand was 200 years. At period 2, 4, and 6, three age groups of stands remain. However, at period 10, no gap in age distribution was observed. Figure 3.13 is for Barker's (1989) forest over 10 periods. The gap in age distribution at period 1 disappears after period 4. At

a)



b)

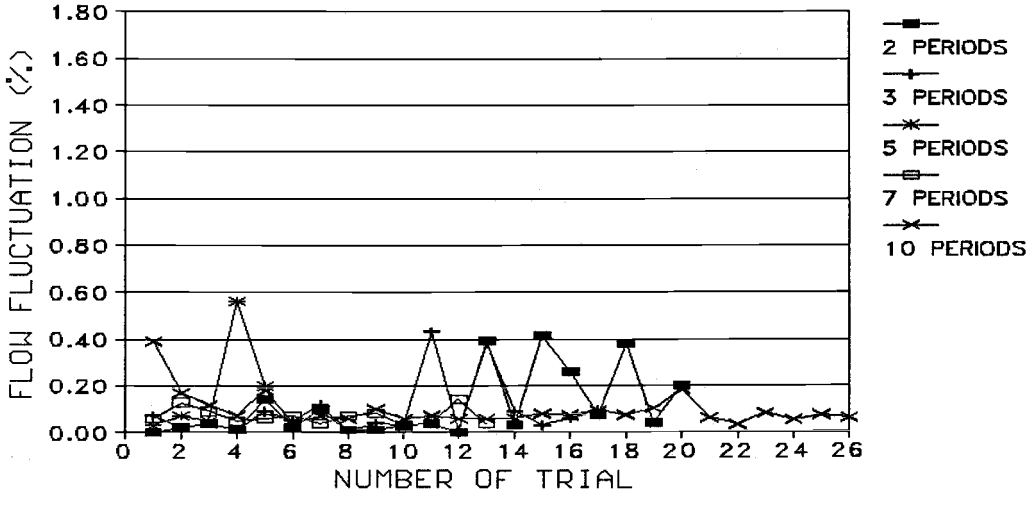
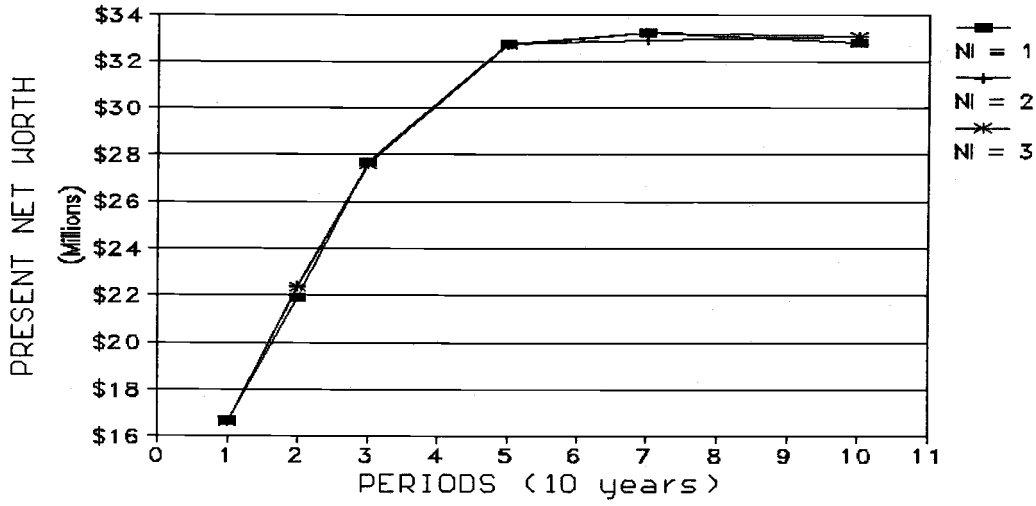


Figure 3.10. Changing patterns of flow fluctuation for the second example forest by SSMART
a) NI = 1
b) NI = 3

a)



b)

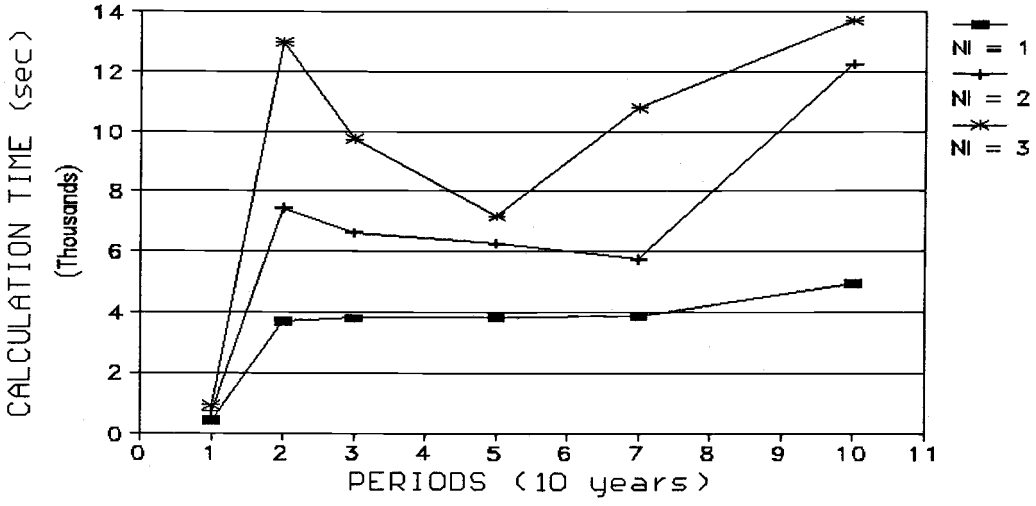


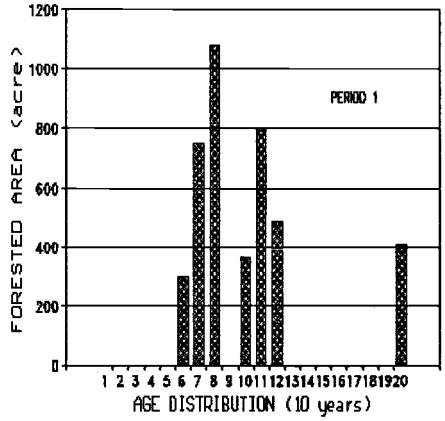
Figure 3.11. Comparison of solutions by SSMART for the second example forest
a) In terms of the objective value
b) In terms of the computational time

Table 3.3. The final solutions for the second example forest by SSMART

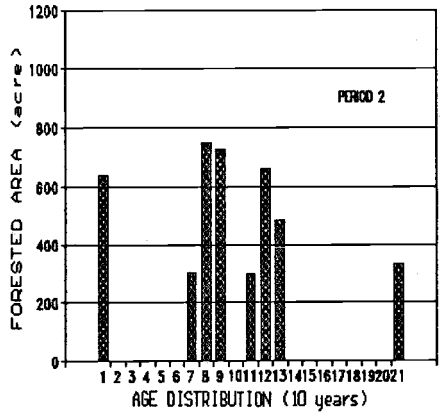
1) Period	2) # of Integer Vars.		1 Internal Iteration	2 Internal Iterations	3 Internal Iterations	Best Solution
1	45	Objective Value	16672700	16626770	16611460	16672700
		Fluctuation (%)	N/A	N/A	N/A	N/A
		Time (second)	404	719	932	404
2	90	Objective Value	21876402	22378024	22328280	22378024
		Fluctuation (%)	0.03	0.09	0.20	0.09
		Time (second)	3656	7417	12981	7417
3	135	Objective Value	27697024	27639188	27584592	27697024
		Fluctuation (%)	0.18	0.14	0.09	0.18
		Time (second)	3793	6601	9737	3793
5	225	Objective Value	32716462	32709842	32697972	32716462
		Fluctuation (%)	0.49	0.33	0.20	0.49
		Time (second)	3816	6237	7156	3816
7	360	Objective Value	33231870	32912416	33211696	33231870
		Fluctuation (%)	0.33	0.08	0.04	0.33
		Time (second)	3844	5727	10773	3844
10	900	Objective Value	32806306	33075206	33064266	33075206
		Fluctuation (%)	0.31	0.09	0.06	0.09
		Time (second)	4914	12236	13697	12236

1): the number of periods
2): the number of integer variables

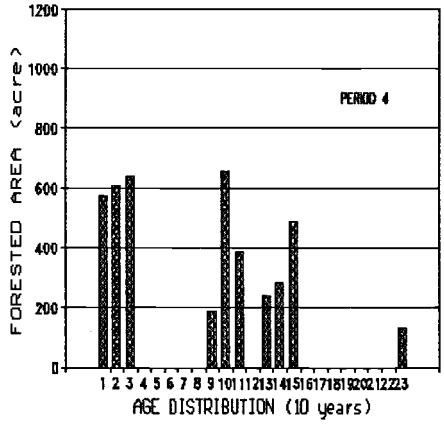
Period 1



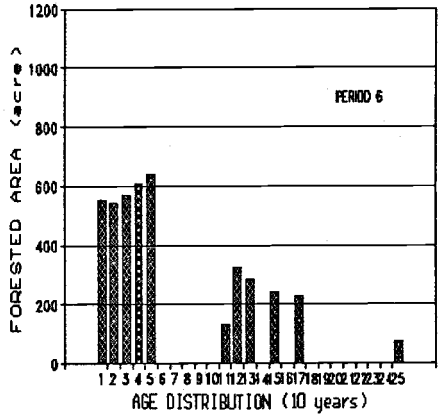
Period 2



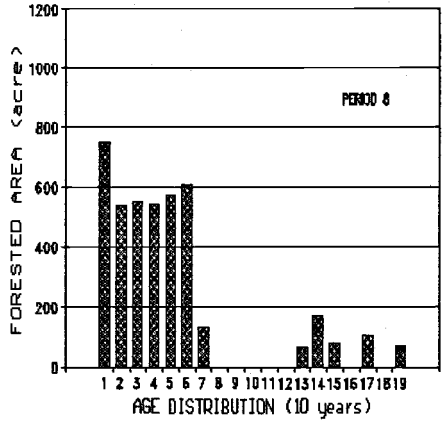
Period 4



Period 6



Period 8



Period 10

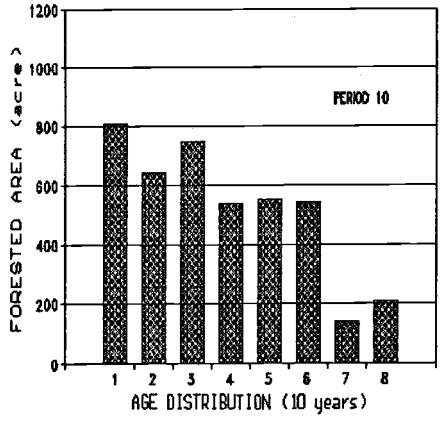
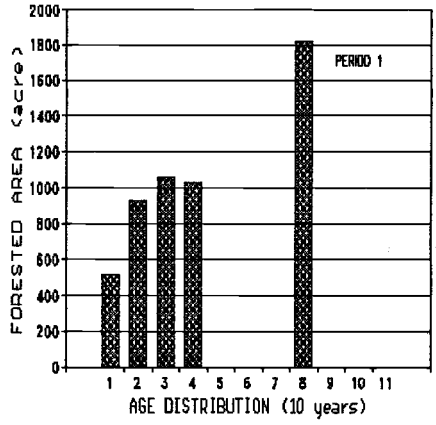
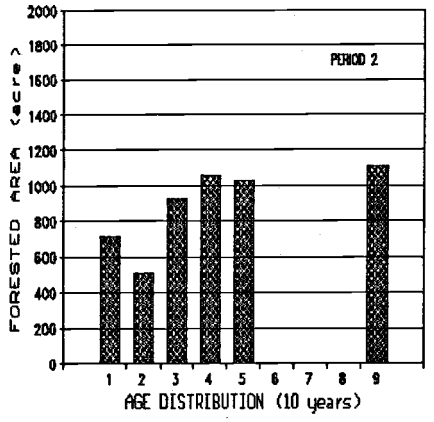


Figure 3.12. Dynamics of stand age distribution of the first example forest over 10 periods by SSMART

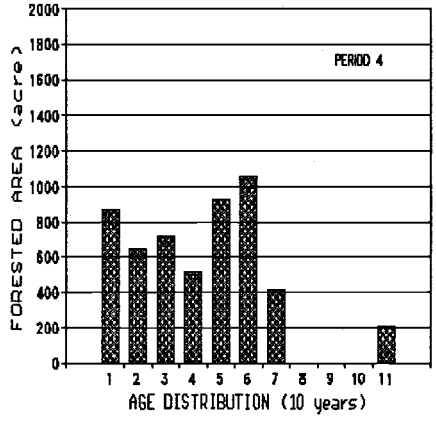
Period 1



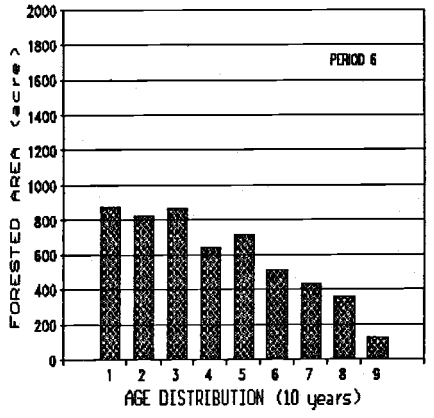
Period 2



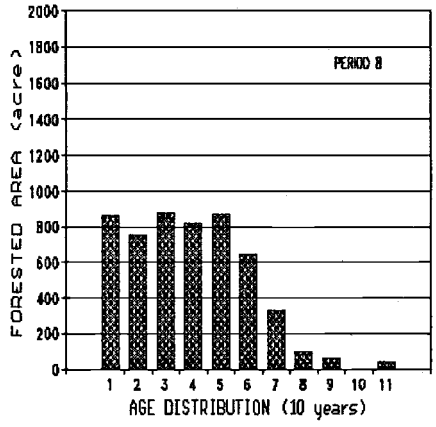
Period 4



Period 6



Period 8



Period 10

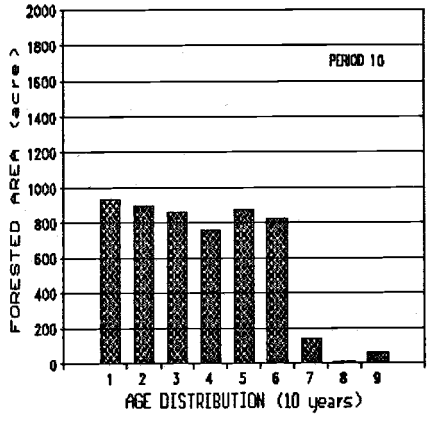


Figure 3.13. Dynamics of stand age distribution of the second example forest over 10 periods by SSMART

period 10, harvest can be implemented from most of stands at age 60, which is the minimum rotation age. Conversion of these unregulated forests into nearly regulated forests, i.e., equal area in each age class, results from the combination of the present net worth objective, minimum harvest age and harvest flow constraints and occurs for longer time horizons without specific inventory constraints.

EVALUATION OF THE PERFORMANCE OF THE PROPOSED ALGORITHM

Two solution techniques were used to compare the performance of the ROHO-PATH algorithm by SSMART. The first utilized commercial software, implementing the branch-and-bound algorithm with LP relaxation. This was accomplished by using LINGO/386 (Language for INteractive General Optimization) (Cunningham and Schrage 1989). The second is the composite relaxation technique. A computer program called RELAX (Torres and Brodie 1989) was used. RELAX was written in FORTRAN 77 code using RM/FORTRAN compiler version 2.43. A Compaq 386/25MHz personal computer with 80387 math-coprocessor was used for calculation.

Before evaluation, the best problem specification for B&B and for RELAX were determined, because using an inefficient formulation for both techniques might result in a tremendous amount of time to find any feasible solution, leading to a false superiority of SSMART. Using the forest from Nelson et al. (1990), selection of the best formulation was completed first.

Solutions By The Branch-and-Bound Algorithm

In order to compare the performance of B&B to others, it is necessary to select the best IP formulation on the basis of the computational time and the final objective

value. For efficiency of B&B varies depending upon the IP formulation structure. Yoshimoto and Brodie (1990) used a single period spatially constrained scheduling problem to conclude that the conventional pairwise adjacency constraint formulation would find a final solution faster than other formulations even if it has the largest number of constraints.

Expanding the time horizon further, land accounting constraints should be taken into account, which might affect the number of constraints and therefore the computational time. Four different IP formulation structures were used. Two different stopping rules for B&B are applied in LINGO/386. One is to use the IP tolerance, IPTOL. Once B&B finds an incumbent integer feasible solution with the objective value, z, the search is restricted to branches which are at least IPTOL times z better than the incumbent. The other is to limit the number of iterations, i.e., pivots. Using these two rules separately or jointly, we can avoid inefficient searching by B&B. Even if B&B finds the optimal solution, it continues searching until it is guaranteed that the solution is optimal. Since heuristic techniques cannot guarantee the optimality of the solution, it is not necessary for B&B to guarantee it for the comparison purpose, either.

Four different IP formulations shared the same objective, maximizing the present net worth of return from

each decision, and the following two sets of constraints,

1. harvest flow constraints
2. 0-1 integer restriction.

With a preferable flow fluctuation, P' = 0.05, even-flow constraints (2) in the original IP formulation were replaced by,

$$\underline{\mathbf{V}} \cdot \underline{\mathbf{X}} \leq (1 + P') \cdot \underline{\mathbf{V0}} \qquad (25)$$

$$\underline{\mathbf{V}} \cdot \underline{\mathbf{X}} \leq (1 - P') \cdot \underline{\mathbf{V0}} \qquad (26)$$

"No activity" for a harvest unit is not considered as one decision.

The first formulation, called CONV, was built with the conventional pairwise adjacency constraints and the land accounting constraints as well as the above two sets of constraints. The number of constraints for the 1-, 2-, 3-, 5-, 7-, and 10-period problems was 85, 215, 300, 470, 1050, and 8275, respectively. Sudden change in number for the 7- and 10-period problems was caused by regeneration harvest activities. Due to the limitation of the number of constraints of LINGO/386, the 10-period problem could not be solved.

For the problem formulated by CONV, two experiments with different stopping rules for the iterative procedure were conducted. The first achievement was done only with 5 % IP tolerance. Results are shown in the fourth column of Table 3.4. Solutions for the 1-, 2- and 5-period problems turned out to be optimal in a sense that there is no

Table 3.4. The final solutions for the first example forest by LINGO/386

1) Period	2) # of Integer Vars.		3) TRAD. IPTOL: 5 %	4) TRAD. PIVOTS: 100000	5) TAM + LA PIVOTS: 100000	6) AUG.TAM PIVOTS: 100000	7) RTAM + LA PIVOTS: 100000
1	45	# of Constraints	85	85	46	46	31
		LP Objective Value	34593898	34593898	36775621	36775621	39412172
		IP Objective Value	30048438	30048438	30048438	30048438	* 30048438
		Time (second)	33	32	457	457	4551
2	90	# of Constraints	215	215	137	94	107
		LP Objective Value	58942449	58942449	59119738	59596727	59976727
		IP Objective Value	46414288	47215532	* 45824392	* 43474552	* 40511072
		Time (second)	355	1055	4943	4406	5041
3	135	# of Constraints	300	300	183	142	138
		LP Objective Value	53005586	53005586	53100039	77003016	53184406
		IP Objective Value	* 50240580	* 52574848	* 45715312	* 47345248	* 35521616
		Time (second)	13756	12397	6096	5162	4627
5	225	# of Constraints	470	470	275	238	200
		LP Objective Value	42818711	42818711	42915129	98418781	42955789
		IP Objective Value	41852676	* 41258700	* 35323932	***	* 30174108
		Time (second)	9461	11068	6666	6737	5018
7	360	# of Constraints	1050	1050	410	379	290
		LP Objective Value	38030715	38030715	38048469	122165063	38054055
		IP Objective Value	* 35922504	* 33975016	* 24251412	***	***
		Time (second)	over 48 hours	15635	8192	16428	7297
10	900	# of Constraints	8275	8275	935	928	635
		LP Objective Value	**	**	35547160	183959797	35553766
		IP Objective Value	**	**	***	***	***
		Time (second)	**	**	18581	41328	13117

1): the number of periods
2): the number of integer variables
3): based on the traditional formulation with the IP tolerance 5%
4): based on the traditional formulation with 100,000 maximum number of pivots
5): based on the triangular adjacency matrix and land accounting constraints
6): based on the augmented triangular adjacency matrix
7): based on the reduced triangular adjacency matrix and land accounting constraints
* : solution may be nonoptimal
**: out of the memory to run the model by LINGO/386
***: no integer solution found

feasible solution 5 % better than the best solution already found. For the other two solutions, there was no guarantee of optimality, so that the computational time to finish iteration was large. However, the number of iterations to find both solutions was 1569 for the 3-period problem and 38983 for the 7-period problem.

To look for the more precise solution, the second test was conducted without the 5 % IP tolerance, but with 100,000 as the limitation of the number of iterations. Results are in the fifth column of Table 3.4. Within the limitation, LINGO/386 found the optimal solution for the 1- and 2-period problems. There was no guarantee of the optimality for the solution of the 3-, 5- and 7-period problems. While the solution of the 3-period problem is better than the solution with 5 % IP tolerance, the opposite is true for the 5- and 7-period problems. For large problems, a large number of iterations is required, so that the IP tolerance stopping rule might be more efficient to find a better IP solution than the other stopping rule, the limitation of the number of iterations. For the rest of the formulations, the limitation 100,000 of the number of iterations was applied for all problems. Also except for the 1-period problem, the 5 % IP tolerance stopping rule is used jointly.

Utilizing Yoshimoto and Brodie (1990), four different adjacency constraints can be derived from one adjacency matrix. Among them, two types of constraints were used for

construction of the adjacency constraints. One is based on the triangular adjacency matrix (TAM), which is made of the upper or lower triangular matrix of the original adjacency matrix. The other is based on the reduced triangular adjacency matrix (RTAM). RTAM is constructed by reducing redundant columns and elements from the adjacency matrix in a sense that there is no duplication in the matrix for elements to represent the adjacency relationships among units.

The second IP formulation, TAM+LA, had the land accounting constraints and used TAM to construct the adjacency constraints. The number of constraints for the second formulation was vastly diminished, 46, 137, 183, 275, 410 and 935 for the 1-, 2-, 3-, 5-, 7-, and 10-period problems, respectively. Remarkable reduction is observed for the 10-period problem. An optimal solution was found for the 1-period problem only. No integer feasible solution was found for the 10-period problem within 100,000 iterations. Results are on the sixth column of Table 3.4.

The third IP formulation, AUG.TAM, utilized the augmented adjacency matrix in TAM. Land accounting constraints were incorporated into the original adjacency matrix. Utilizing TAM for the augmented adjacency matrix, the formulation had 46, 94, 142, 238, 379 and 928 constraints for the 1-, 2-, 3-, 5-, 7-, and 10-period problems respectively. An optimal solution was found for

the 1-period problem, while no integer feasible solution was found for the 5-, 7- and 10-period problems as shown in Table 3.4.

The last formulation, RTAM+LA, consists of land accounting constraints and adjacency constraints derived from RTAM of the original adjacency matrix as well as harvest flow constraints and 0-1 integer restriction. The number of constraints was further reduced except for the 2-period problem. There was no guarantee for optimality of the solution for all problems. The solution of the 1-period problem, however, was the same as the optimal solution found by other formulations. No integer feasible solutions were found for the 7- and 10-period problems.

Across different formulations, CONV had the largest number of constraints followed by TAM+LA, then AUG.TAM and RTAM+LA. AUG.TAM had fewer constraints for the 2-period problem than RTAM+LA, while for other problems, RTAM+LA had fewer constraints than AUG.TAM. On the basis of the objective value of the LP relaxed solution, CONV provided the smallest value followed by TAM+LA. AUG.TAM provided the second largest value for the 1- and 2-period problems, however, for other problems, it provided the largest value. Since the objective value of the LP relaxed solution is always greater than the IP solution, it can be said that CONV provides the closest LP relaxed solution to the IP solution. As for the IP solutions, CONV provided the best

solutions for all problems except the 10-period problem in terms of the objective value. Although TAM+LA provided an inferior IP solution for the 3-period problem to the solution by AUG.TAM, TAM+LA may be the second best in considering the use of 5 % IP tolerance. The objective value of the solution for the 3-period problem by AUG.TAM was not 5 % better than the solution by TAM+LA. The IP solutions by RTAM+LA were inferior to others.

Solutions By The Composite Relaxation Technique

For the use of the composite relaxation technique, a computer problem RELAX was used. RELAX has a capacity to handle 450 integer variables and up to 250 adjacency constraints. Land accounting constraints are implicitly handled. Since the model uses "no activity" as one treatment to all harvest units, the number of integer variables is always greater than that of the IP formulation by the number of harvest units. Because of the use of the subgradient method in dealing with the even-flow constraints, problems with 2 or more periods may be solved.

Four different specifications of adjacency constrains were used in order to seek the best solutions by the composite relaxation technique. They were derived from Yoshimoto and Brodie (1990). The first adjacency constraints were based on the original adjacency matrix

(OAM), the second were based on TAM, the third were based on the reduced adjacency matrix (RAM), and the last ones were based on RTAM. None of them was augmented to include land accounting constraints. The objective value, percentage of flow fluctuation, and computational time of each solution are shown at Table 3.5.

For the 2-period problem, the first formulation based on OAM provided the highest objective value with the smallest percentage of flow fluctuation. The fourth formulation by RTAM had the second highest objective value, followed by the third by RAM and the second by TAM. For the 3-period problem, however, the second formulation provided the highest objective value, followed by the first, the fourth and the third formulation.

Applying 5 % preferable flow fluctuation, only the solution by the first formulation was feasible for the 5-period problem. Others had more than 5 % flow fluctuation with less objective value.

Due to the limitation of the number of adjacency constraints on RELAX, the first and second formulations with 360 and 344 constraints were unable to be solved at the 7-period problem. As for the solution by the fourth formulation, there was no integer solution after 20 hours of running RELAX. Only the third formulation had a solution. However, percentage of flow fluctuation was out of the preferable range.

Table 3.5. The final solutions for the first example forest by RELAX

1) Period	2) # of Integer Vars.		3) OAM	4) TAM	5) RAM	6) RTAM
2	135	Objective Value	44363879	39611121	41980926	43505996
		Fluctuation (%)	0.02	0.09	3.31	0.15
		Time (second)	2420	2660	1319	1590
3	225	Objective Value	44000969	46787496	42779445	43484223
		Fluctuation (%)	0.88	2.16	3.71	2.36
		Time (second)	10165	11126	7085	6309
5	360	Objective Value	39805277	36950094	37364219	38577133
		Fluctuation (%)	2.31	17.50	27.87	17.6
		Time (second)	15883	15763	10347	11413
7	405	Objective Value	**	**	28417299	***
		Fluctuation (%)	**	**	40.99	***
		Time (second)	**	**	16411	over 20 hours
10	945	Objective Value	**	**	**	**
		Fluctuation (%)	**	**	**	**
		Time (second)	**	**	**	**

1): the number of periods
2): the number of integer variables
3): based on the original adjacency matrix
4): based on the triangular adjacency matrix
5): based on the reduced adjacency matrix
6): based on the reduced triangular adjacency matrix
**: out of the limitation of the model RELAX
***: no integer solution found

In terms of the computational time, the first and the second formulations seem to take more time than the others. Since searching the best surrogate multipliers is most likely affected by the number of adjacency constraints, the less constraints, the less computational time.

Comparison Of Different Techniques

Comparison of different techniques for spatially constrained area-based harvest scheduling problem has been completed. Techniques were the B&B algorithm by LINGO/386, the composite relaxation technique by RELAX, and the ROHO-PATH algorithm by SSMART. One hundred thousand iterations were used as the limitation for all problems with LINGO/386. Five percent IP tolerance was jointly used in the 5- and 7-period problems.

The first comparison was done using Nelson et al.'s (1990) forest. Results are depicted in Figure 3.14. Two solutions for LINGO/386 were provided. The first solution had 5 % preferable flow fluctuation, and the other has P % preferable flow fluctuation given by the solutions derived from SSMART. Although 5 % is preferable, solutions from SSMART had flow fluctuation less than 5 %. Thus, it is fair to compare the solutions given the same flow fluctuation. For all techniques, the best solutions in terms of the objective value were selected.

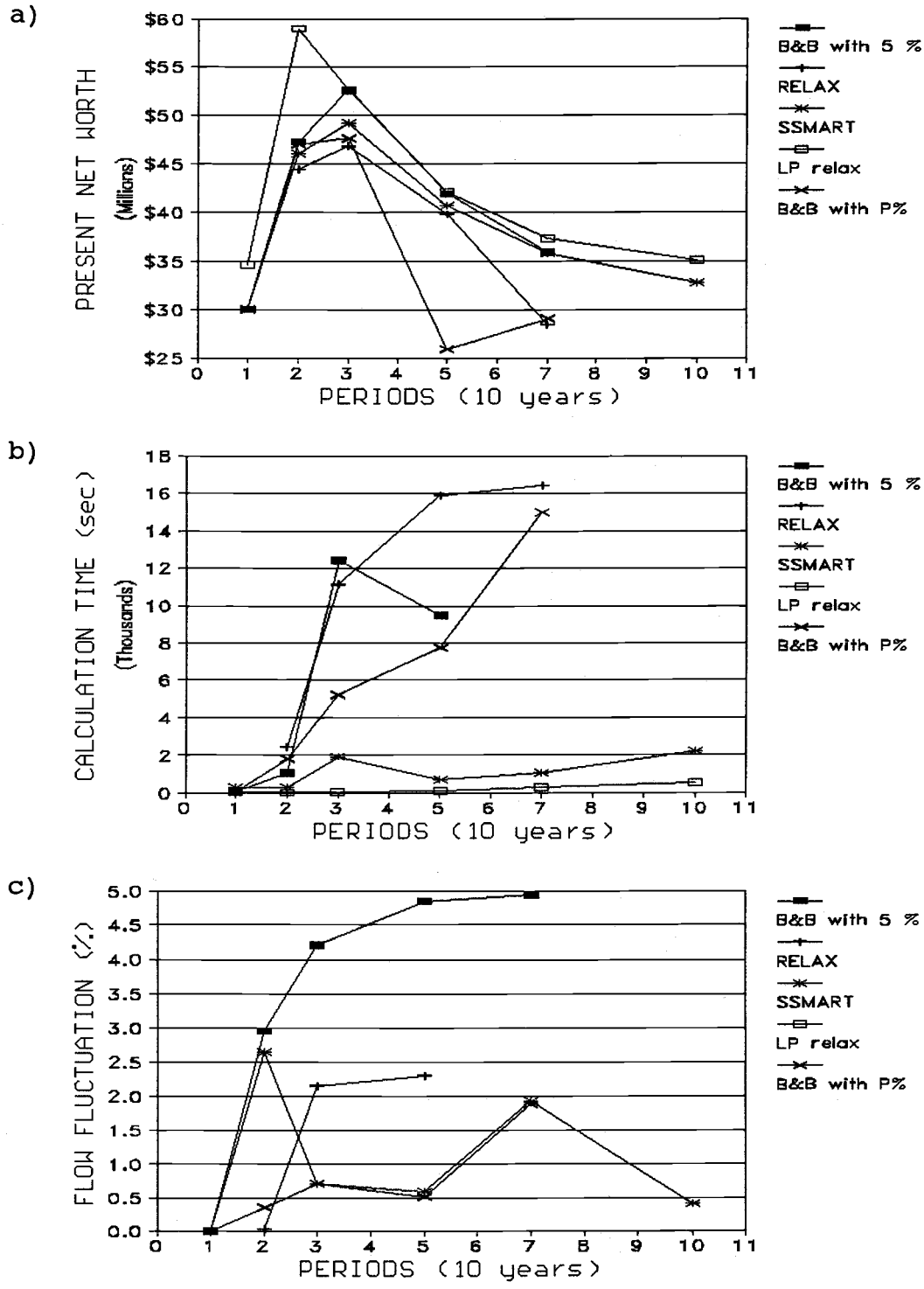


Figure 3.14. Comparison of techniques for the first example forest
a) In terms of the objective value
b) In terms of the computational time
c) In terms of the flow fluctuation

In terms of the objective value (Figure 3.14-a), LINGO/386 with 5 % flow fluctuation yielded the highest value for all problems except the 10-period problem. The difference among solutions from LINGO/386 with 5 % and SSMART was 2.6 %, 6.5 %, 2.9 %, and 0.7 % for the 2-, 3-, 5- and 7-period problems, respectively. For the 1-period problems the solutions were the same for both techniques. As opposed to RELAX, SSMART yielded higher objective value by 3.5 %, 4.9 %, and 2.1 % for the 2-, 3-, and 5-period problems. Comparing the objective value of solutions by SSMART and LINGO/386 with P %, LINGO/386 yielded a higher solution for the 2-period problem by 2 %, while SSMART provided better solutions for the rest of the problems, 3-, 5-, and 7-period problems. Although the solution for the 3-period problem by LINGO/386 was 3.3 % less than that of SSMART, others were 36.2 % (5-period) and 18.6 % (7-period) less than those by SSMART. This is due to the fact that solutions by LINGO/386 may not be optimal. The LP relaxation with P % flow fluctuation yielded 13.1 %, 22.0 %, 6.3 %, 3.3 %, 4.3 % and 3.6 % higher objective value for the 1- to 10-period problems than those by SSMART. It is observable that for the larger problem, the solutions by SSMART are close to those of the LP relaxation in terms of the objective value.

Figure 3.14-b describes the computational time. Because of the use of the limitation of the number of

iterations for LINGO/386, the computational time for LINGO/386 did not increase exponentially. Except for the 1-period problems, the computational time by SSMART was less than the others. For the 5-and 7-period problems, it was less than one-tenth of those by LINGO/386. RELAX yielded the largest computational time for the 2-, 5-, and 7-period problems and the second largest for the 3-period problems.

As for the quality of the final solution, i.e., actual flow fluctuation, it is depicted in Figure 3.14-c. Since 5 % flow fluctuation was used for LINGO/386 with 5 %, the flow fluctuation of LINGO/386 with 5 % was close to 5 %. RELAX yielded the smallest flow fluctuation for the 2-period problem. The other solutions by RELAX, however, had more fluctuation than SSMART. Using P % given by SSMART, LINGO/386 yielded smaller fluctuation for the 2-period problem than that of SSMART, and almost the same for other problems. Since the LP relaxation with P % had the same fluctuation as those by SSMART, they were not depicted in Figure 3.14-c.

The second comparison using a forest from Barker (1989) was completed between SSMART and LINGO/386 with P % only. The objective value, computational time, and actual flow fluctuation are depicted in Figure 3.15. Compared to the solutions from LINGO/386, SSMART provided better solutions except for the 1-period problem in terms of the objective value. For the 1-period problem, LINGO/386 had 1.3 % higher

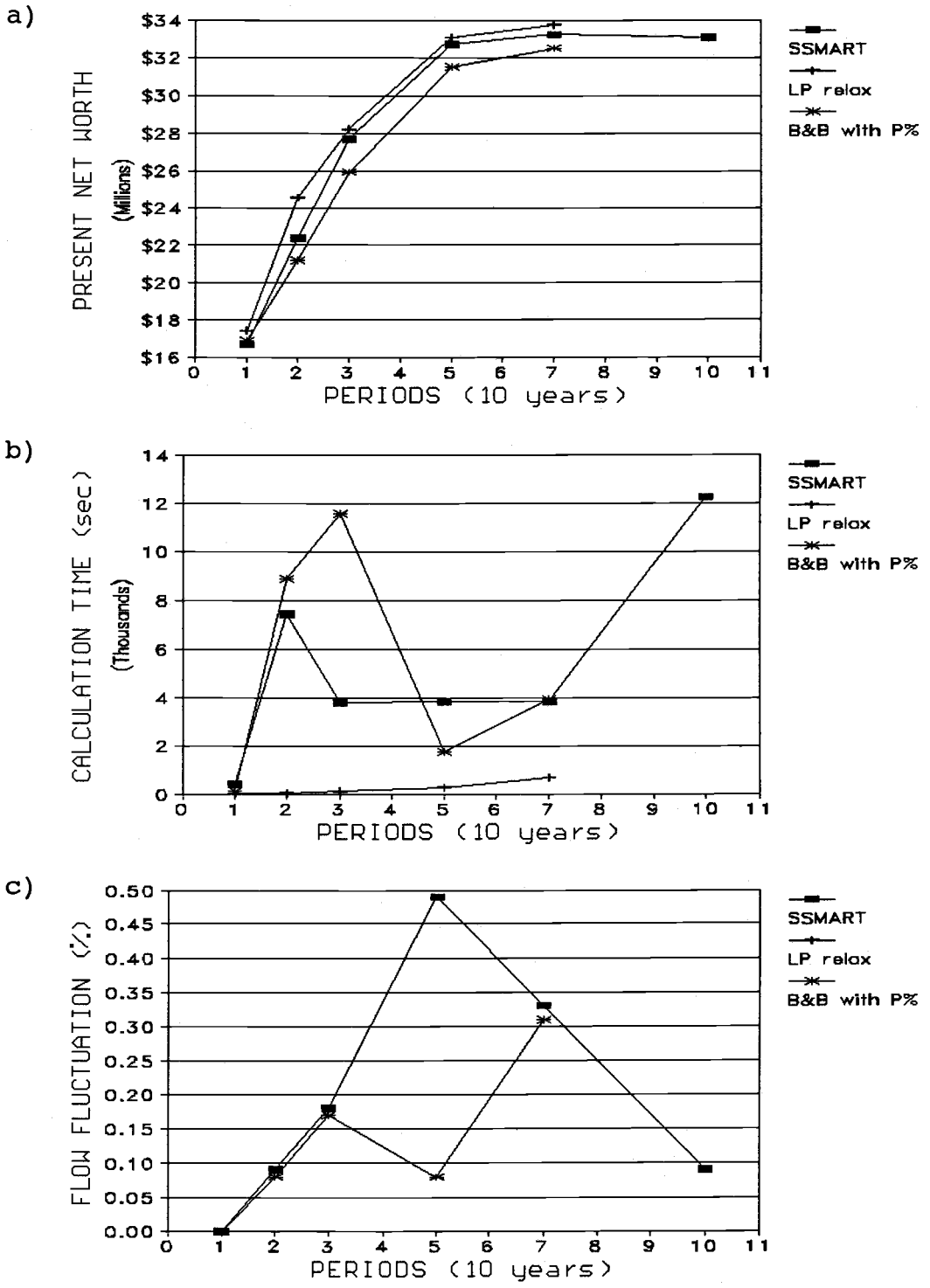


Figure 3.15. Comparison of techniques for the second example forest
a) In terms of the objective value
b) In terms of the computational time
c) In terms of the flow fluctuation

objective value, but 5.4 %, 6.4 %, 3.6 %, and 2.2 % lower for the 2-, 3-, 5-, and 7-period problems than those by SSMART. Due to the limitation of the memory in LINGO/386, the 10-period problem was not solved by B&B. Solutions for the 5- and 7-period problems turned out to be optimal with 5 % IP tolerance. Because of this, the computational time for them was relatively small as opposed to others. When 5 % IP tolerance was removed to find another solution, the computational time increased almost ten times as much as before. The objective value of a new solution for the 5-period problem was better than the previous one, while it was not so for the 7-period problem. The objective value of both problems were still lower than those by SSMART (Table 3.6).

Table 3.6. Comparison of solutions by SSMART and LINGO/386

1) Period		Nelson et al.'s (1990) forest			Barker's (1989) forest		
		2) ROHO-PATH	3) LP relax with P%	4) B&B with P %	ROHO-PATH	LP relax with P%	B&B with P %
1	Objective Value	30048438	34595898	30048438	16672700	17423051	16887100
	Fluctuation (%)	N/A	N/A	N/A	N/A	N/A	N/A
	Time (second)	312	10	32	404	20	121
2	Objective Value	45985504	58942449	46965004	22378024	24566182	* 21175700
	Fluctuation (%)	2.64	2.64	0.35	0.09	0.10	0.08
	Time (second)	299	25	1791	7417	49	8876
3	Objective Value	49174844	52466539	* 47552324	27697024	28206467	* 25922600
	Fluctuation (%)	0.71	0.72	0.71	0.18	0.19	0.17
	Time (second)	1909	56	5218	3793	104	11575
5	Objective Value	40659432	42041422	* 25924420	32716462	33060594	31534800 (*30086400)
	Fluctuation (%)	0.59	0.60	0.52	0.49	0.50	0.08 (0.50)
	Time (second)	670	98	7742	3816	272	1763 (17028)
7	Objective Value	35680836	37279965	* 29035244	33231870	33731809	32498000 (*32522900)
	Fluctuation (%)	1.94	1.95	1.88	0.33	0.34	0.31 (0.28)
	Time (second)	1016	286	14978	3844	676	3937 (33067)
10	Objective Value	32767306	33979398	***	33075206	**	**
	Fluctuation (%)	0.41	0.42	***	0.09	**	**
	Time (second)	2171	443	18812	12236	**	**

1): the number of periods
2): the ROHO-PATH Algorithm
3): LP relaxation with P fluctuation % given by ROHO-PATH
4): the Branch-and-Bound Algorithm with P fluctuation % given by ROHO-PATH
* : solution may be nonoptimal
**: out of the limitation of the model RELAX
***: no integer solution found
( ): without 5 % IPTOL

CONCLUSIONS

Due to the difficulty of solving the spatially constrained area-based harvest scheduling problem by the integer programming technique, several heuristics have been developed. The main purpose of this paper was to propose another heuristic based on the random search technique.

The commonly used random search techniques (Sessions and Sessions 1988, O'Hara et al. 1989, Nelson et al. 1990) utilize the random number to be assigned to each harvest unit so that each unit is determined to be harvested if the corresponding random number is greater than or equal to a certain value. By chance, therefore, ineffectual iteration can occur in such cases because the number of selected unit combinations is too small to satisfy the harvest flow constraints due to failure to select some viable candidates. The random search technique (ROHO) in the proposed model, on the other hand, utilizes the random number to make an ordered sequence of harvest units. Given the descending ordered sequence, units are selected for harvest in order, not randomly. As a result, at each iteration, none of the candidates for harvest is missed unless the harvestable volume by selected units exceeds the given minimum harvest flow level.

Another difference between ROHO and other random search techniques is that the solution derived at each period by

ROHO has a "minimum" harvest flow fluctuation, while the solution by the others is only guaranteed to be feasible. In addition, the solution at each period by ROHO is feasible at that period and the following, while the solution by the others is only feasible at the present period except in the case of Nelson et al. (1990).

Combining the ROHO algorithm and the PATH algorithm (ROHO-PATH), the model called SSMART was developed. The solutions by SSMART were insensitive to change in the maximum number of internal iterations in terms of the objective value. However, it can be expected that the more the maximum number of internal iterations, the less the actual flow fluctuation, and the longer the computational time.

As observed in Yoshimoto and Brodie (1990) for a single period scheduling problem, the conventional pairwise adjacency constraints were efficient for the branch-and-bound algorithm for multiperiod scheduling problems as long as the number of constraints is less than the limitation of commercial software, e.g., LINGO/386. For RELAX, the accuracy of the solution seems to depend on specification of the adjacency constraints.

Because of the use of each decision (treatment) as one integer variable, the number of integer variables for the IP formulation and RELAX increases as the number of regenerated harvests increases. This is not the case for SSMART. Since

the PATH algorithm is incorporated into SSMART, the number of integer variables is the same as the number of harvest units times the number of periods. It is not affected by the number of regenerated harvest units.

Comparing the performance of LINGO/386 and RELAX, SSMART appears to yield a "good" feasible solution for all period problems with a reasonable computational time and fairly small flow fluctuation. With an infinite amount of time, LINGO/386 could find an optimal solution, while with 100,000 iterations, the solutions for the 5 or more period problems were inferior to those by SSMART. Since the longer the time horizon, the more the integer variables, and thus, the more the iterations needed, superiority of SSMART over LINGO/386 might be obvious for long-term problems.

Although SSMART did not deal with road network optimization, it is possible to combine SSMART with a heuristic road network optimization by Sessions (1987) without difficulty (Sessions and Sessions 1988).

The advantage of the proposed algorithm results from partitioning the problem period by period, then respecifying the objective of the subproblem at each period by minimizing flow fluctuation. As a result, the final solution has not only the largest objective value but also small flow fluctuation among a generated set of feasible solutions.

LITERATURE CITED

Arney, J.D. 1985. A modeling strategy for the growth projection of managed stands. Can. J. For. Res. 15:511-518.

Barker, B.R. 1989. Utilizing scheduling and network analysis program (SNAP) as a forest plan implementation tool on the Siuslaw National Forest. M.S. thesis, College of Forestry, Oregon State Univ., Corvallis, OR. 76p.

Berck, P. and T. Bible. 1984. Solving and interpreting large-scale harvest scheduling problems by duality and decomposition. For. Sci. 30:173-182.

Brooke, S.H. 1958. A discussion of random methods for seeking maxima. Opns. Res. 6:244-251.

Bullard, H.S., H.D. Sherali, and W.D. Klemperer. 1985. Estimating optimal thinning and rotation for mixed-species timber stands using a random search algorithm. For. Sci. 31:303-315.

Conley, W. 1980. Computer optimization techniques. Petrocelli Books Inc., New York. 266p.

Cunningham, K. and L. Schrage. 1989. The LINGO modeling language. LINDO Systems Inc., Chicago, IL. 93p.

Dakin, R.J. 1965. A tree search algorithm for mixed integer programming problems. Computer J. 8:250-255.

Dantzig, G.B. 1951. Maximization of a linear function of variables subject to linear inequalities. Chap. 21 in:

Activity Analysis of Production and Allocation, (T.C. Koopmans, ed.) Cowles Commission Monograph No. 13, Wiley, New York.

Gavish, B. and H. Pirkul. 1985a. Zero-one integer programs with few constraints - lower bounding theory. Europ. J. Opns. Res. 21:213-224.

Gavish, B. and H. Pirkul. 1985b. Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality. Math. Prog. 31:78-105.

Geoffrion, A.M. 1974. Lagrangean relaxation for integer programming, in:M.L. Balinski, ed., Math. Prog. Study No.2 (North-Holland, Amsterdam):82-114.

Ghandforoush, P. and B.J. Greber. 1986. Solving allocation and scheduling problems inherent in forest resource management using mixed-integer programming. Comput. & Opns. Res. 13:551-562.

Glover, F. 1975. Surrogate constraint duality in mathematical programming. Opns. Res. 23:434-453

Greenberg, H.J. and W.P. Pierskalla. 1970. Surrogate mathematical programming. Opns. Res. 18:924-939.

Held, M., P. Wolfe, and H.P. Crowder. 1974. Validation of subgradient optimization. Math. Prog. 6:62-88.

Hoganson, H.M. and D.W. Rose. 1984. A simulation approach for optimal timber management scheduling. For. Sci. 30:220-238.

Johnson, K.N. and T.W. Stuart. 1987. FORPLAN version 2:

Mathematical programmer's guide. USDA For. Serv., Land Management Planning Systems Section, Washington, DC. 158p.

Kirby, M.W., P. Wong, W.A. Hager, and M.E. Huddleston. 1980. A guide to the integrated resources planning model. USDA For. Serv., Berkeley, CA. 211p.

Land, A.H. and A.G. Doig. 1960. An automatic method for solving discrete programming problems. Econometrica 28:497-520.

Meneghin, B.J., M.W. Kirby, and J.G. Jones. 1988. An algorithm for writing adjacency constraints efficiently in linear programming models. The 1988 Symp. on Systems Analysis in Forest Resources. USDA For. Serv. Gen. Tech. Rep. RM-161:46-53. Rocky Mountain Forest and Range Exp. Stn., Fort Collins, CO.

Navon, D. 1971. Timber RAM - a long-range planning method for commercial timber lands under multiple use management. USDA For. Serv. Res. Paper PSW-70. Pacific Southwest Forest and Range Exp. Stn., Berkeley, CA. 22p.

Nelson, J., J.D. Brodie, and J. Sessions. 1990. Integrating short-term, area-based logging plans with long-term harvest schedules. For. Sci. Monograph. In journal review.

Nelson, J. 1988. Integration of short-term spatially feasible harvesting plans with long-term harvest

schedules using Monte-Carlo integer programming and linear programming. Ph.D. Dissertation, Dept. of Forest Management, Oregon State Univ., Corvallis, OR. 168p.

Nemhauser, G.L. and L.A. Wolsey. 1988. Integer and combinatorial optimization. John Wiley & Sons Inc. New York. 763p.

O'Hara, A.J., B.H. Faaland, and B.B. Bare. 1989. Spatially constrained timber harvest scheduling. Can. J. For. Res. 19:715-724.

Paredes V., G.L. and J.D. Brodie. 1987. Efficient specification and solution of the even-aged rotation and thinning problem. For. Sci. 33:14-29.

Schrage, L. 1979. A more portable Fortran random number generator. ACM Trans. Math. Softw. 5:132-138.

Sessions, J. 1987. A heuristic algorithm for the solution of the variable and fixed cost transportation problem. In Proc: The 1985 Symp. on System Analysis in Forest Resources. Univ. of Georgia, Athens, GA. pp.324-336.

Sessions, J. and J.B. Sessions. 1988. SNAP - a scheduling and network analysis program for tactical harvest planning. In Proceedings of International Mountain Logging and Pacific Northwest Skyline Symp., Dec. 12-16 1988. Oregon State Univ., Corvallis, OR. pp.71-75.

Torres R., J.M. and J.D. Brodie. 1990. Adjacency constraints in harvest scheduling: an aggregation heuristic. Can. J. For. Res. In press.

Torres R., J.M. and J.D. Brodie. 1989. Manual to run program RELAX. Unpublished manuscript.

Torres R., J.M., J.D. Brodie, and J. Sessions. 1990a. Solution to the area-based harvest scheduling problem through Lagrangean relaxation. In journal review.

Torres R., J.M., J.D. Brodie, and J. Sessions. 1990b. The use of relaxation to solve the habitat dispersion problem. In journal review.

Yoshimoto, A., G.L. Paredes V., and J.D. Brodie. 1988. Efficient optimization of an individual tree growth model. The 1988 Symp. on Systems Analysis in Forest Resources. USDA For. Serv. Gen. Tech. Rep. RM-161:154-162. Rocky Mountain Forest and Range Exp. Stn., Fort Collins, CO.

Yoshimoto, A., R.G. Haight, and J.D. Brodie. 1990. A comparison of the pattern search algorithm and the modified PATH algorithm for optimizing an individual tree model. For. Sci. In press.

Yoshimoto, A. and J.D. Brodie. 1990. Comparative efficiency of algorithms to generate adjacency constraints. In journal review.

Zanakis, S.H. and J.R. Evans. 1981. Heuristic "optimization":why, when, and how to use it. Interfaces 11:84-91.

Chapter 4

COST EVALUATION OF IMPLEMENTING SPATIAL RESTRICTIONS ON RIPARIAN ZONE PLANNING IN WESTERN OREGON

by

Atsushi Yoshimoto

and

J. Douglas Brodie

ABSTRACT

Due to the significance of shade, wildlife habitat, soil stabilization, and water-filtering effects of stream side vegetation on the riparian zone, spatial restrictions are often imposed by state and federal agencies on both public and private lands. From the forest owners' point of view, any restriction could reduce profit. In order to satisfy the public needs, and minimize the forest owners' costs resulting from new restrictions, appropriate regulations are proposed. A case study was performed on the Green River Subbasin on the Alsea Ranger District, Siuslaw National Forest in Oregon using three scenarios in order to estimate the cost of implementing different types of spatial restrictions. In this study, prohibiting harvest in the riparian management area cost approximately $14,954.59 to

$17,951.73 per foot of zone width from the stream in the example forest. Allowing harvest in the riparian management area with additional adjacency constraints among segments of the riparian management area cost almost nothing to $3,439.73 for the basin per foot width of the riparian management area. Imposing an additional adjacency lag period as a restriction resulted in a marginal cost per lag period of $2,266,219 to $2,820,074 for the whole subbasin.

INTRODUCTION

The framework of the forest level problem is to allocate land and existing resources over the time horizon to meet a physical or economic objective function subject to a range of multiple-resource constraints. Among many multiple-use resource allocation problems, planning for multiple-resource production from riparian zones has received a great deal of attention over the last decade both politically and economically.

Elmore and Beschta (1987) summarized multiple-use of riparian areas. Riparian areas provide forage for domestic animals and important habitat for wildlife species. They provide essential habitat for resident fish and other aquatic organisms if streams are perennial. When overbank flows occur, riparian areas can attenuate flood peaks and increase groundwater recharge. The character and condition of riparian vegetation and associated stream channels influence property values. Aesthetics and water quality are also important.

Social concerns and values with regard to the products, services and factors involved in the production processes from forest lands and the riparian zone have been translated into new acts, laws, and regulations. In 1987, the State Board of Forestry in Oregon developed new regulations to protect riparian zones and incorporated them into the Oregon

Forest Practice Rules. According to the Oregon Forest Practice Rules (1988), it is required that any landowner, operator, or timber owner conducting an operation shall retain a riparian management area along each side of Class I waters. Restrictions on operations in riparian zones provide an environment for evaluation of costs for alternate policies which may eventually, when combined with estimates of benefits of riparian zoning, provide justification for relaxation or tightening restrictions in future legislation.

A case study on riparian zone planning was done by Olsen et al. (1987) before the revision of the forest practice rules. They estimated the landowners' costs by using three rule scenarios and concentrating on impacts of harvest systems and harvest returns. They concluded that landowners' cost would be significantly greater than costs under the old rules.

There was no specific harvest scheduling in their work. Instead they used the present net worth of the cash flow impacts spread out equally across the planning periods. Due to the lack of precise information on the spatial and dynamic characteristics of the harvest schedule, their estimates would differ from those that would be provided by an optimization technique applied to old and new rule scenarios.

Mathematical programming techniques currently are often utilized to provide harvest schedules for both public and

private land. Timber Resource Allocation Model (RAM) (Navon 1971) and FORPLAN (Johnson and Crim 1986, Johnson et al. 1986) are widely used linear programming based models. When utilizing a linear programming based model, a decision variable can be any nonnegative real value. Due to dimensionality limitations, formulations must be highly aggregated for applied scale problems. As a result, disaggregation of a solution is required before a forest manager can implement it on the ground.

Disaggregation of a solution can be avoided if a decision variable is forced to be dichotomous, 0-1 integer variable. The problem with 0-1 integer variables can be solved by integer programming or mixed integer programming. The Integrated Resources Planning Model (IRPM) by Kirby et al. (1980) is designed for these integer problems.

Due to the limitations of the integer programming technique with respect to the number of integer variables and constraints, alternative heuristic techniques have been developed (Sessions and Sessions 1988, O'Hara et al. 1989, Nelson et al. 1990, Torres et al. 1990, Yoshimoto et al. 1990). Heuristic techniques are designed to provide a "good" solution without a guarantee of optimality. Heuristics provide for faster solution of smaller integer problems and can provide solution to larger problems that cannot be formulated or solved using integer and mixed-integer programming.

The objective of this paper is to estimate the cost of implementing spatial regulations on the riparian zone in the context of implementing spatially constrained harvest scheduling with even-flow constraints. No costs for alternative harvest systems, e.g., yarding, skidding and road network construction, are considered. A heuristic technique developed by Yoshimoto et al. (1990) is used to search for a "good" harvest schedule. Changes in the width of riparian zone protected and in harvest sequences allowed, are evaluated. In the next section, the harvest scheduling model by Yoshimoto et al. (1990) is reviewed. Then, the model is applied to a study area in western Oregon and the economic impact of alternate strategies is evaluated.

THE SCHEDULING SYSTEM OF MANAGEMENT ALTERNATIVES FOR TIMBER-HARVEST (SSMART)

This paper presents a new method for solving the spatially constrained area-based harvest scheduling problem with even-flow constraints. The objective of the problem is to maximize present net worth of the sum of returns from each stand. Adjacency constraints are imbedded in order to avoid large clearcut areas, and to provide habitat for wildlife. In addition, even-flow constraints are utilized. The decision variables are set up in the same way as in the Model I formulation (Johnson and Stuart 1987), and are restricted to be 0-1 integer, so that a forest manager can apply a solution without disaggregation.

Defining the decision variable by,

$$x_i = \begin{cases} 1 \text{ if the i-th decision is implemented,} \\ 0 \text{ otherwise} \end{cases}$$

the proposed problem is formulated by the following integer programming formulation,

$$Z_{IP} = \underset{\{\underline{X}\}}{\text{maximize}}\ \underline{C}^t \cdot \underline{X} \tag{1}$$

subject to

$$\underline{M} \cdot \underline{X} \leq \underline{M0} \tag{2}$$

$$\underline{V} \cdot \underline{X} = \underline{V0} \tag{3}$$

$$\underline{L} \cdot \underline{X} \leq \underline{1} \tag{4}$$

$$\underline{X} \in \{0, 1\}$$

where $\underline{C}$ is a coefficient vector representing present net

worth of return from a decision vector $\underline{X} = (x_1, x_2, , , , x_N)^t$ and N is the number of decisions. "t" denotes the transpose. Equation (2) is the adjacency constraint. Derivation of a matrix $\underline{M}$ and a vector $\underline{M0}$ can be found in Meneghin et al. (1988), Torres and Brodie (1990), or Yoshimoto and Brodie (1990). The conventional pairwise formulation can also be applied to equation (2). Equation (3) is the even-flow constraint. A matrix $\underline{V}$ consists of elements, v_{ij}, which is a harvestable volume contributed by the i-th decision at the j-th period. $\underline{V0}$ is a vector to ensure an even-flow level. Equation (4) is the land accounting constraint, which implies that at most one decision can be implemented for one harvest unit.

Exact solution techniques, i.e., the integer programming solution techniques, are impractical in terms of the computational time and storage capacity necessary to handle normal applied problems. In addition, minimally infeasible solutions can be found for the problem specifications which are inherently infeasible and are sufficient for applied management. Some infeasibility is inherent in the even-flow constraint (equation (3)). Other desired conditions may not initially exist in long-term multiperiod problems.

The Scheduling System of Management Alternatives foR Timber-harvest (SSMART) by Yoshimoto et al. (1990) is one of several heuristic techniques proposed to solve a spatially

constrained area-based harvest scheduling problem with even-flow constraints over long time horizons of up to 20 periods. SSMART utilizes the ROHO-PATH algorithm (Random Ordering Heuristic Optimization with Projection Alternative TecHnique) in order to make the iterative procedure efficient and provide a "good" solution. To overcome the inherent infeasibility of a problem with even-flow constraints, the problem in SSMART is respecified by not only maximizing the present net worth of the sum of returns from decisions, but also minimizing harvest flow deviation,

$$Z_{IP} = \underset{\{\underline{X}\}}{\text{maximize}}\ \underline{C}^t \cdot \underline{X} = \Sigma \underline{C}_i^t \cdot \underline{X}_i \qquad (1)$$

where $\underline{X}_i$ is the solution at the i-th period of the following subproblem,

$$Z_{IP}(i) = \underset{\{\underline{X}_i\}}{\text{minimize}}\ |\underline{V}_i \cdot \underline{X}_i - V1| \qquad (5)$$

subject to

$$\underline{M}_i \cdot \underline{X}_i \leq \underline{M0}_i \qquad (6)$$

$$\underline{M}_{i+1} \cdot \underline{X}_{i+1} \leq \underline{M0}_{i+1} \qquad (7)$$

$$(1-P) \cdot V0 \leq \underline{V}_i \cdot \underline{X}_i \leq (1+P) \cdot V0 \qquad (8)$$

$$(1-P) \cdot V0 \leq \underline{V}_{i+1} \cdot \underline{X}_{i+1} \leq (1+P) \cdot V0 \qquad (9)$$

$$V1 = (1-P) \cdot V0 \qquad (10)$$

$$\underline{X}_i, \underline{X}_{i+1} \in \{0, 1\}$$

where $\underline{M}_i$ is a submatrix of $\underline{M}$, $\underline{V}_i$ is a submatrix of $\underline{V}$, $\underline{M0}_i$ is a subvector of $\underline{M0}$, for the corresponding $\underline{X}_i$, respectively. The land accounting constraint is handled implicitly. P is a user-specified acceptable flow fluctuation.

The modified random search algorithm called ROHO is used to generate a solution for the subproblem $Z_{IP}(i)$ at the i-th period. The commonly used random search technique (Sessions and Sessions 1988, O'Hara et al. 1989, Nelson et al. 1990) is such that a random number is assigned to each harvest unit so that it is harvested if the assigned random number is greater than or equal to a certain value. Thus, choice of a harvest unit heavily depends on its random number. By contrast, the ROHO algorithm assigns a random number to each harvest unit. The stochastic nature of a random search technique is only used to create the descending ordered sequence of units based on its random number. Selecting a harvest unit depends on this ordered sequence, not its random number. Harvest units adjacent to the previously selected units are eliminated as candidates for a solution at the same time. By changing the ordered sequence, a set of feasible solutions for the above subproblem is created. The final solution of the subproblem is selected with the minimum harvest flow deviation from V1 among feasible solutions at each period. Since feasibility of a generated solution should be guaranteed at both the current and the following periods, the same procedure to generate a feasible solution at the immediately following period is applied whenever a feasible solution is found at the current period.

The binary search method with modification is used to

obtain an optimal or appropriate even-flow level, V0, and the final solution for the problem. Unlike other random search techniques by Sessions and Sessions (1988), O'Hara et al. (1989) and Nelson et al. (1990), the solution derived from SSMART has not only the largest objective value over the time horizon, but also the minimum flow deviation among a generated set of feasible solutions at each period.

In order to run SSMART, three main parameters are to be specified by the user. These values are used to make the iterative process in SSMART efficient. They are:

1. Maximum number of internal iterations: NI,
2. Maximum number of entire iterations over the time horizon: NE,
3. Maximum number of sequential failures: NF.

To select a final solution, the user provides two different levels of flow fluctuation. The first one is the acceptable flow fluctuation, P, and the second is the preferable flow fluctuation, P' (<P). A set of feasible solutions satisfies the P flow fluctuation, and the final solution is selected from a restricted set of feasible solutions satisfying P' flow fluctuation. Further description of SSMART can be found in Yoshimoto et al. (1990).

SITE SELECTION AND DATA DESCRIPTION

A case study is conducted on the Green River Subbasin on the Alsea Ranger District, Siuslaw National Forest in Oregon (Barker 1989). The total forested area is 6238 acres including 893 acres in private blocks. A forest map showing harvest blocks and the riparian system is depicted in Figure 4.1. There are two age groups of stands. One is mature stands over age 80. The other is young growth stands. The stand age of young growth stands ranges from 10 to 40 years. Stand age distribution at the current period is given in Figure 4.2. One hundred stands out of 137 non-private stands are adjacent to a stream. Among these 100 stands, 46 are mature stands. Species is assumed to be Douglas-fir (*Pseudotsuga menziesii* (Mirb) Franco).

Volume yields for these stands were derived using a single-tree/distance-independent growth model called SPS (Stand Projection System) by Arney (1985). The stand conditions for the 10 year age class and subsequent stands developed by SPS are given in Table 4.1. The site index is 135 feet on a breast-height age basis. Using this stand, volume yields at age 80, and from age 10 to age 40 were generated. Future volume of each stand was also predicted by SPS.

Timber values were calculated using a price equation. The price equation is a function of a diameter at breast

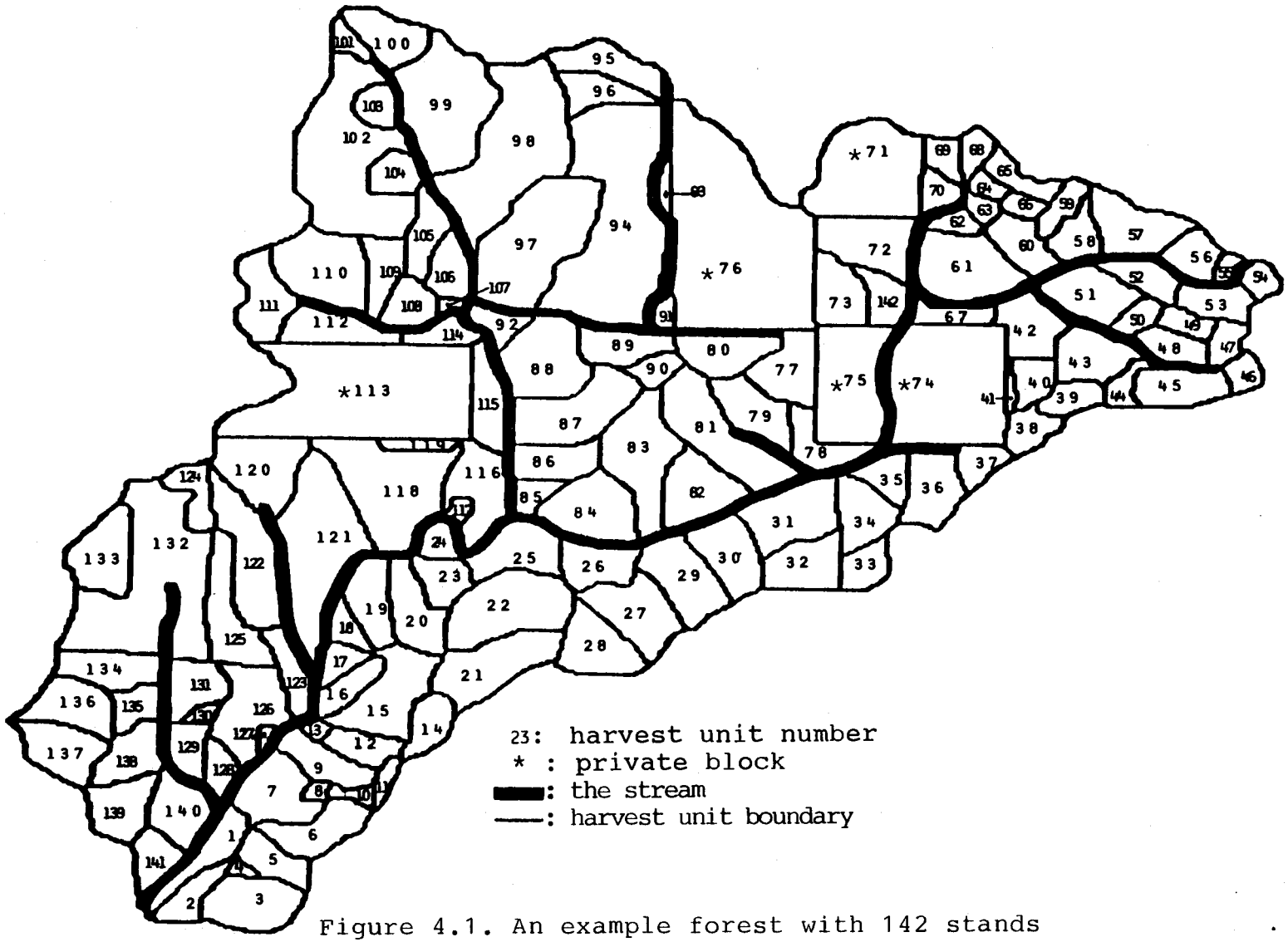


Figure 4.1. An example forest with 142 stands

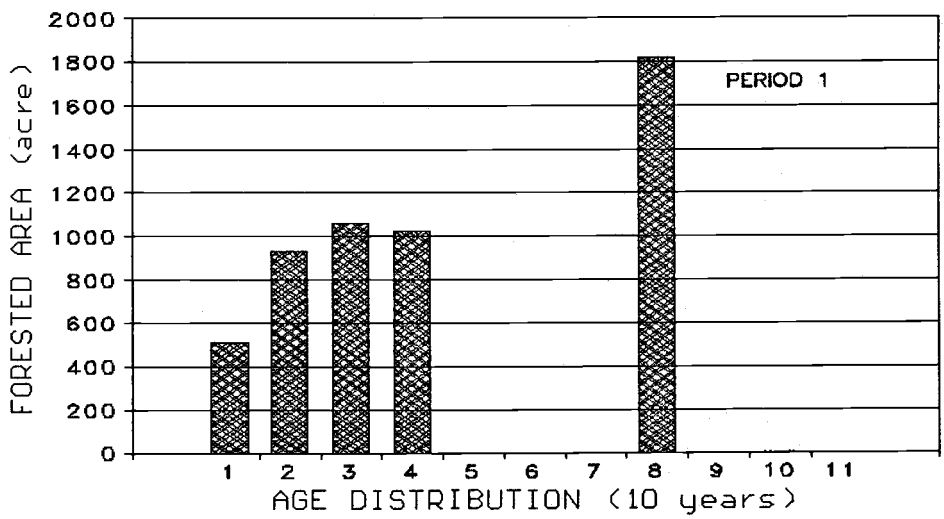


Figure 4.2. Stand age distribution for an example forest

Table 4.1. Initial stand condition for SPS (see Arney 1985)

		SP	SITE	AGE	REGION			
STAND		DF	135	10	PNW			
MERCH		1.0	16.4	4.0	6.0			
NAME		input	data	file	for	SPS		
		SP	DBH	TOP	TPA	AGE	SDEV	NAT
TABLE	1	DF	1.0	30.0	100.	10	.25	1
TABLE	2	DF	2.0	32.0	150.	10	.25	1
TABLE	3	DF	3.0	33.0	50.	10	.25	1
CLUMP		.90						
REPORTS		10	20	30	40	50	60	80

height (dbh) on the basis of thousand merchantable board feet. Price premium was assumed with respect to dbh,

$$\$Price/mbf = 12.5 \cdot dbh \qquad (11)$$

As shown in Table 4.1, the minimum dbh for a merchantable tree is 6 inches, and the top diameter limit is 4 inches. For the smallest diameter tree, the stumpage value is $75.00/mbf, and for a tree with 20-inch diameter, it is $250.00/mbf. Prices are assumed constant through the time horizon in the analysis, i.e., no real price increases.

A two hundred dollar regeneration cost per acre is embedded into the timber value of each stand. A four percent constant real interest rate was used to calculate the present net worth of return from each stand. Clearcutting is the only option for harvest.

COST EVALUATION OF SPATIAL RESTRICTIONS

As mentioned before, the Oregon Forest Practice Rules (1988) protect an aquatic area by establishing riparian management areas, or buffers, along an aquatic area. The width of the riparian management area may vary dependent upon topography, vegetative cover, the needs of harvesting design, and the needs for aquatic and wildlife habitat. For streams, the width of the riparian management area shall average three times the stream width, but it shall not average less than 25 feet or more than 100 feet. Stream width is the average of the main channel width of the stream during its high water level flow. Figure 4.3 shows the relationship between the aquatic area and the riparian management area.

In this analysis, the width of the riparian management area is assumed constant throughout the stream length. The cost of implementing spatial restrictions on the riparian zone is evaluated in terms of the difference in the present net worth from the solution without any spatial restrictions:

$$COST(\$PNW) = \$PNW_{\text{without restrictions}} - \$PNW_{\text{with restrictions}} \quad (12)$$

Three scenarios were proposed for cost evaluation. The first scenario (called the static-scenario) was modeled according to the Oregon Forest Practice Rules. That is, no harvest was allowed within the riparian management area.

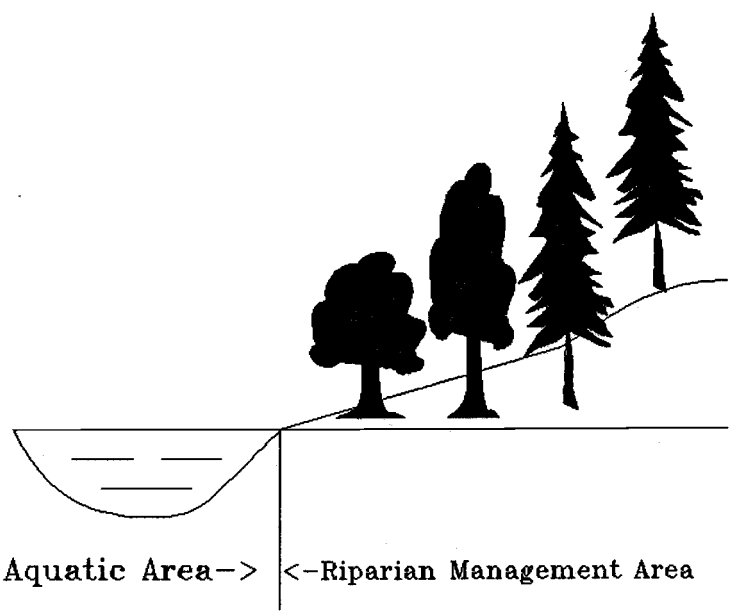


Figure 4.3. The relationship between an aquatic area and the riparian management area

In the second scenario (called the dynamic-scenario), harvest was allowed in the riparian management area. However, adjacency restrictions among segments of the riparian management area were applied. In other words, the riparian management area specified in each harvest unit, if any, is treated as another harvest unit. Adjacency restrictions are
used among all units including segments connected by the stream. This scenario can be used to provide wildlife with shelter on at least one side of the stream through the stream length.

The third scenario (called the lag-scenario) applied a different number of periods for the adjacency lag, during which no adjacent harvest occurs. The more periods for the adjacency lag, the larger the forested area protected along the stream at each period. No additional riparian management is imposed for this scenario.

For the first two scenarios, four different buffer sizes were used, 50, 100, 150, and 200 feet over four different time horizons, 3, 5, 7, and 10 periods. One period is set as 10 years. The adjacency lag was set as 1 period. In the third scenario, 2 and 3 periods were used for the adjacency lag over four different time horizons, 3 to 10 periods. In using SSMART, each parameter was specified as follows:

NI = 1 and 2,

NE = 10,

NF = 5,

P = 10 (%),

P' = 5 (%).

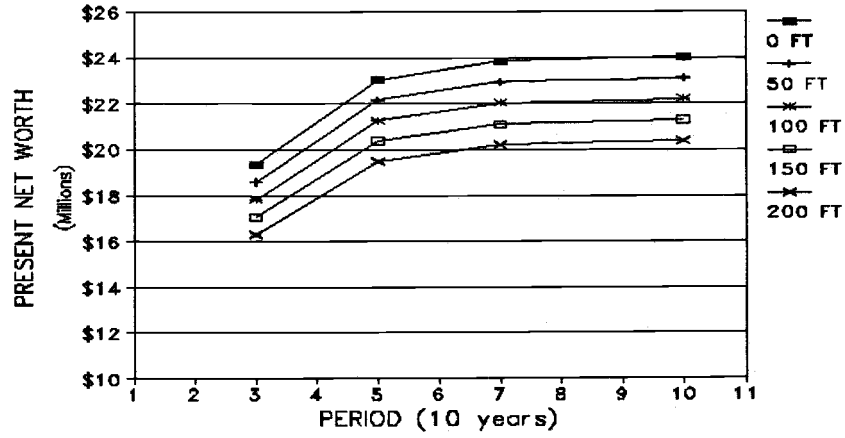
As can be seen, two different trials with different NI's are implemented to search for the best solution. Minimum rotation age was set as 60 years (6 periods). Thus no regenerated harvests occur for the problems with the 3- and the 5-period time horizon.

Static Change In Buffer Size

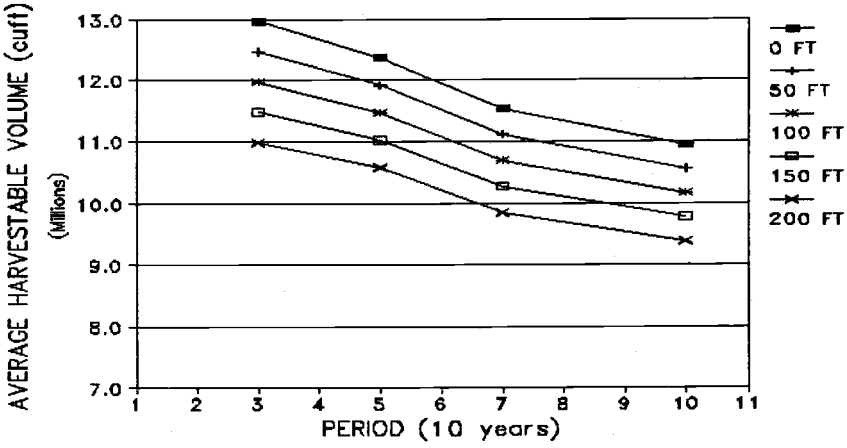
For the static-scenario based on the Oregon Forest Practice Rules (1988), two approaches are used to estimate the cost of implementing spatial restrictions on the riparian management area. The first approach utilized SSMART to obtain solutions without any restrictions. Then subtracting harvestable volume and returns of the riparian management area from the solution, a new solution for each restriction is derived. The second approach applied SSMART to all problems with or without spatial restrictions on the riparian zone. Spatial restrictions thus influenced the solution throughout the search iterations.

Using the first approach, Figure 4.4-a shows the change in the objective value, present net worth, over the time horizon. Without restriction, the objective value increases

a)



b)



c)

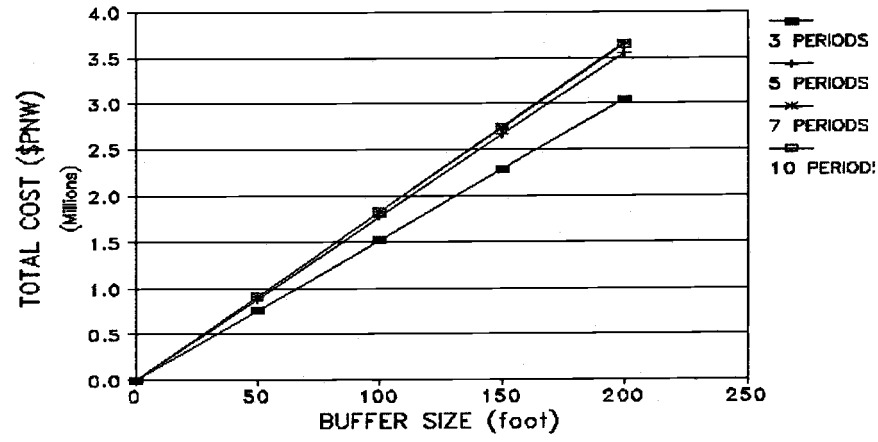


Figure 4.4. Results from the static-scenario by the approach without the use of SSMART to constrained problems
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost

asymptotically toward $24 million from $19 million as the time horizon increases. Prohibiting harvest on the riparian management area, the objective value also goes up asymptotically toward $23 million, $22 million, $21 million, and $20 million for the 50-foot, 100-foot, 150-foot and 200-foot buffer size problems, respectively. Figure 4.4-b shows changing patterns of the average harvestable volume. The longer the time horizon, the less the average harvestable volume. Figure 4.4-c shows the derived cost over buffer size. As can be seen, the cost for the problem with the 3-period time horizon was underestimated compared to the others. Cost for the 200-foot buffer size at the 3-period time horizon was $3 million as opposed to $3.6 million for the 10-period time horizon. In comparison to costs by the 7- and the 10-period time horizon, all costs for different buffer sizes were almost the same. This might suggest that the 3-period time horizon is too short to estimate the cost, while the 7-period time horizon is long enough to approximate the cost. All results from the first approach are given in Table 4.2.

SSMART was also used in the second approach. Changing patterns of the objective value over the different time horizon, the average harvestable volume and the derived costs are depicted in Figure 4.5. Changing patterns of the objective value and the average harvestable volume were almost the same as those in Figure 4.4. In comparison of

Table 4.2. Summary of results from the static-scenario in the first approach
No use of SSMART to constrained problems

a) CHANGE IN OBJECTIVE VALUE

	** 0 FT	50 FT	100 FT	150 FT	200 FT
* 30 yr	19,345,662	18,586,924	17,825,338	17,063,318	16,303,105
50 yr	23,046,212	22,160,335	21,273,294	20,382,515	19,496,645
70 yr	23,848,500	22,937,322	22,022,092	21,104,055	20,190,651
100 yr	24,015,170	23,103,884	22,191,957	21,276,546	20,366,425

b) CHANGE IN AVERAGE HARVESTABLE VOLUME

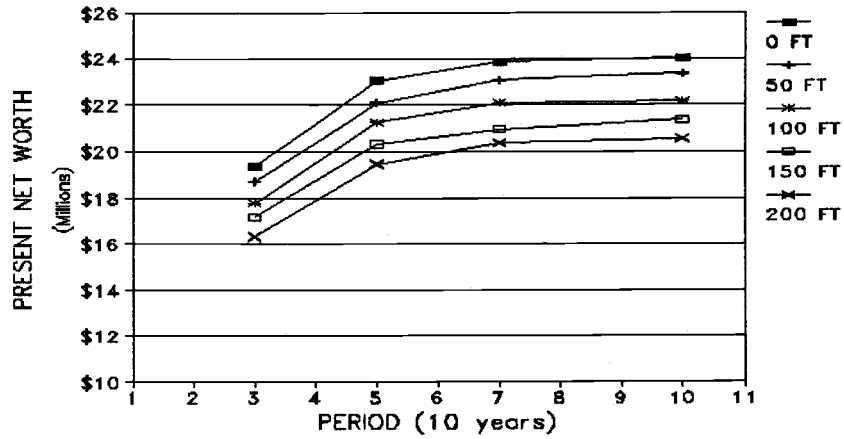
	** 0 FT	50 FT	100 FT	150 FT	200 FT
* 30 yr	12,959,281	12,467,748	11,975,704	11,482,856	10,991,117
50 yr	12,368,536	11,915,629	11,469,640	11,021,138	10,575,559
70 yr	11,528,570	11,109,402	10,692,686	10,272,754	9,856,381
100 yr	10,948,496	10,558,289	10,169,249	9,777,893	9,389,041

c) COST (PNW$)

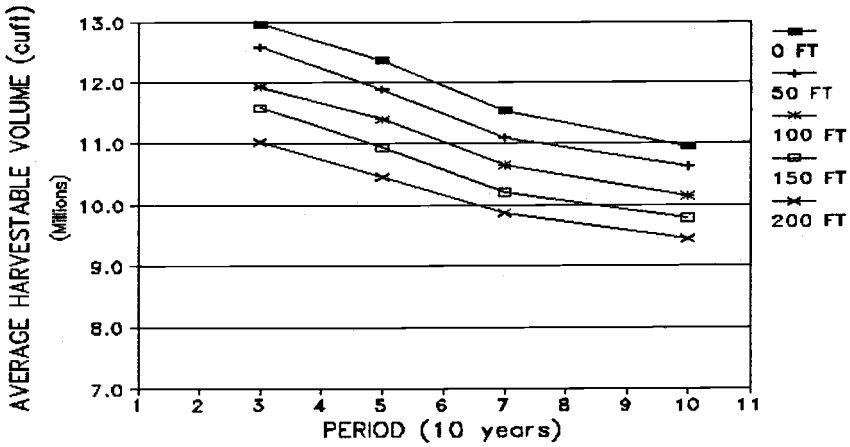
	* 30 yr	50 yr	70 yr	100 yr
** 0 FT	0	0	0	0
50 FT	758,738	885,877	911,178	911,286
100 FT	1,520,324	1,772,918	1,826,408	1,823,213
150 FT	2,282,344	2,663,697	2,744,445	2,738,624
200 FT	3,042,557	3,549,567	3,657,849	3,648,745

* :the time horizon
** : buffer size

a)



b)



c)

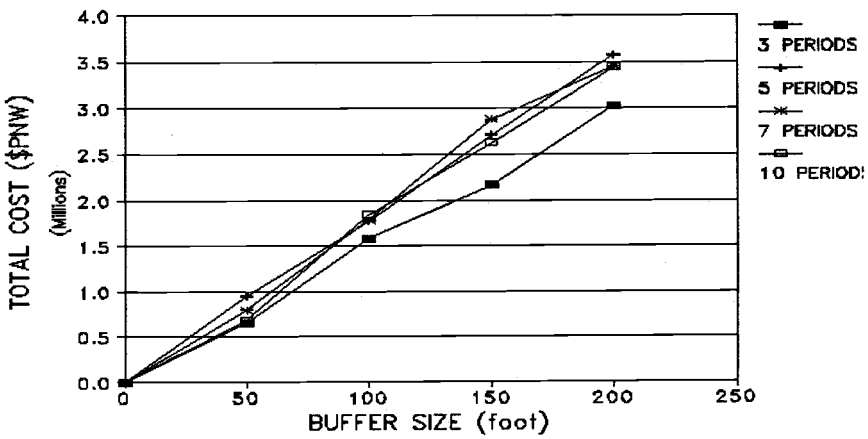


Figure 4.5. Results from the static-scenario by the approach with the use of SSMART to constrained problems
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost

the objective value as shown in Table 4.2 and Table 4.3, the second approach provided slightly higher objective value for 9 problems, the 50-foot, 150-foot and 200-foot buffer size problems of the 3-period time horizon, the 50-foot, 100-foot, and 200-foot buffer size problems of the 7-period time horizon, and the 50-foot, 150-foot, and 200-foot buffer size problems of the 10-period time horizon, while the first approach provided higher objective value for the other 7 problems. The largest difference relative to the absolute objective value can be found for the 200-foot buffer size problem of the 7-period time horizon. The percentage difference was a 1.03 percent increase in the objective value using the second approach. A superiority of the objective value by the first approach was observed in the 150-foot buffer size problem with the 7-period time horizon. It was 0.65 percent higher than the other approach. In terms of the average harvestable volume, however, the second approach provided higher average harvestable volume for only 6 problems, the 50-foot, 150-foot, 200-foot buffer size problems of the 3-period time horizon, the 200-foot buffer size for the 7-period time horizon, and the 50-foot and 200-foot buffer size problems of the 10-period time horizon. The solutions of the 50-foot and 100-foot buffer size problems of the 7-period time horizon and the 150-foot buffer size problem of the 10-period time horizon provided higher objective value with less average harvestable volume

Table 4.3. Summary of results from the static-scenario in the second approach SSMART applied to all constrained problems

a) CHANGE IN OBJECTIVE VALUE

	** 0 FT	50 FT	100 FT	150 FT	200 FT
* 30 yr	19,345,662	18,686,692	17,764,066	17,175,062	16,321,182
50 yr	23,046,212	22,094,390	21,271,996	20,334,390	19,473,246
70 yr	23,848,500	23,055,726	22,068,474	20,966,192	20,400,664
100 yr	24,015,170	23,326,596	22,165,508	21,380,644	20,572,250

b) CHANGE IN AVERAGE HARVESTABLE VOLUME

	** 0 FT	50 FT	100 FT	150 FT	200 FT
* 30 yr	12,959,281	12,585,407	11,927,494	11,582,897	11,020,995
50 yr	12,368,536	11,888,273	11,388,368	10,930,197	10,458,278
70 yr	11,528,570	11,090,902	10,641,401	10,200,149	9,875,940
100 yr	10,948,496	10,621,172	10,133,388	9,776,593	9,427,117

c) COST (PNW$)

	* 30 yr	50 yr	70 yr	100 yr
** 0 FT	0	0	0	0
50 FT	658,970	951,822	792,774	688,574
100 FT	1,581,596	1,774,216	1,780,026	1,849,662
150 FT	2,170,600	2,711,822	2,882,308	2,634,526
200 FT	3,024,480	3,572,966	3,447,836	3,442,920

* :the time horizon
** : buffer size

than the first approach.

As for the derived cost over the buffer size, it increases as the buffer size increases. Figure 4.5-c also suggests that the 3-period time horizon is too short, similar to the conclusion reached with Figure 4.4-c.

Although the solutions by these two approaches are almost the same in terms of the objective value, the quality of the solution, i.e., the actual flow fluctuation is not. The actual flow fluctuation, FLUC, is defined by,

$$\text{FLUC}(\%) = \frac{\text{Flow.max} - \text{Flow.min}}{\text{Flow.max} + \text{Flow.min}} \cdot 100 \ (\%) \qquad (13)$$

where Flow.max is the maximum harvest flow and Flow.min is the minimum harvest flow in the solution. Equation (13) implies that harvest flow at each period lies within $\pm$ FLUC (%) from an even-flow level and FLUC is the smallest flow fluctuation to satisfy it.

Table 4.4 shows the actual flow fluctuation of each solution. With the 50-foot buffer size, the first approach without the application of SSMART to the constrained problems, had 1.01 % at the 3-period time horizon as the smallest fluctuation, and 1.42 % at the 7-period time horizon as the largest one, while the second approach applying SSMART to all problems had 0.13 % at the 10-period time horizon as the smallest and had 0.50 % at the 7-period time horizon as the largest. As for the 100-foot buffer size, the first approach provided 2.29 % as the smallest,

Table 4.4. Comparison of the actual flow fluctuation in the static-scenario

	*50 FT		100 FT		150 FT		200 FT	
	SSMART	NO OPT.	SSMART	NO OPT.	SSMART	NO OPT.	SSMART	NO OPT.
**30 yr	0.38	1.01	0.59	2.29	0.06	3.69	0.77	5.20
50 yr	0.30	1.15	0.07	2.45	0.24	3.85	0.33	5.42
70 yr	0.50	1.42	0.34	2.81	0.26	4.30	0.92	5.94
100 yr	0.13	1.20	1.05	2.54	0.30	3.98	0.35	5.52

NO OPT.:no use of SSMART to constrained problems
SSMART:SSMART applied to all constrained problems
* :buffer size
**:the time horizon

2.81 % as the largest. The second approach provided 0.07 % as the smallest and 1.05 % as the largest. Solving the problem with 150-foot buffer size, the first approach provided 3.69 % minimum fluctuation and 4.30 % maximum, while the second approach provided 0.06 % as the minimum and 0.30 % as the maximum fluctuation. With the 200-foot buffer size problem, 5.20 % is the minimum fluctuation and 5.94 % is the maximum for the first approach. By contrast, 0.33 % is the minimum and 0.92 % is the maximum for the second approach by SSMART. As can be seen from this result, the second approach by SSMART provided less actual flow fluctuation for all constrained problems than the first approach without the application of SSMART to the constrained problems. The average actual flow fluctuation by SSMART is 0.41 % with 0.29 % standard deviation, while the average actual flow fluctuation by the first approach is 3.30 % with 1.68 % standard deviation. It is observed in Table 4.4 that the larger the buffer size, the larger the actual flow fluctuation by the first approach. On the other hand, this is not true for the solutions by SSMART.

Dynamic Change In Buffer Relationship

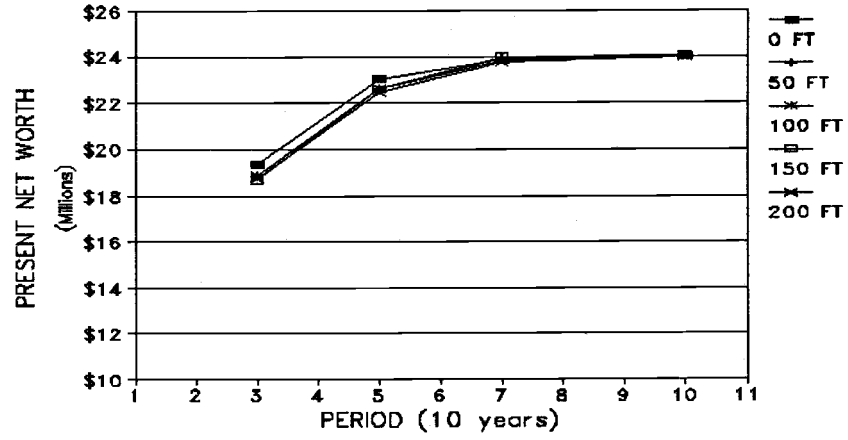
In the dynamic-scenario, a segment of the riparian management area in each harvest unit adjacent to the stream is treated as another harvest unit. The number of

additional harvest units, therefore, is 103 including 3 units from the private blocks. Adjacency constraints are applied among segments in the riparian management area, a segment and its original harvest unit, as well as segments connected by the stream.

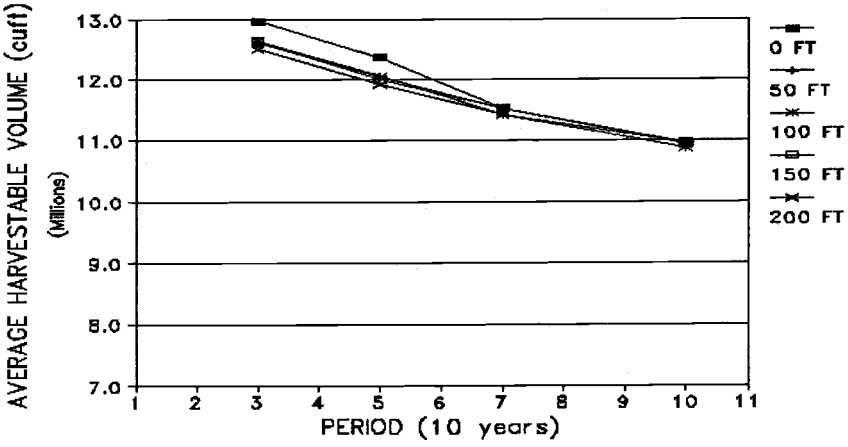
Changing patterns of the objective value, the average harvestable volume, and the derived cost are illustrated in Figure 4.6. Unlike the patterns in Figure 4.4 or 4.5, the objective value of problems with different buffer size converges to $24 million. For the 10-period time horizon, there is almost no difference in the objective value among the 0-, 50-, 100-, 150- and 200-foot buffer size problems, while the objective value of the 50-, 100-, 150-, and 200-foot buffer size problems for the 3-period time horizon is approximately 2 % below the one found in the no restriction problem (Table 4.5).

Because of small differences in the objective value with different buffer size (Table 4.5), the derived cost line is rather flat over the buffer size as can be seen in Figure 4.6-c. Especially for the 7- and the 10-period time horizon, the derived cost was even negative for the 50-foot and 150-foot buffer size problems. For the 3- and the 5-period time horizon, the derived cost of the 50- to 200-foot buffer size problems were almost the same, $4.8 million average with $.7 million standard deviation.

a)



b)



c)

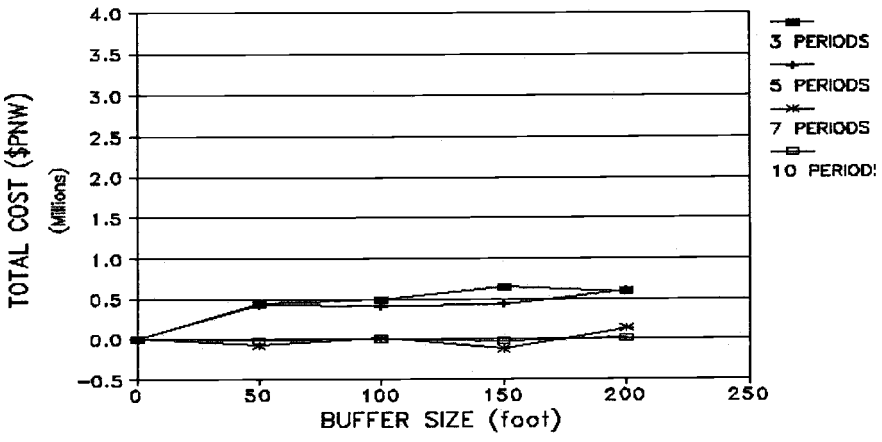


Figure 4.6. Results from the dynamic-scenario
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost

Table 4.5. Summary of results from the dynamic-scenario

a) CHANGE IN OBJECTIVE VALUE

	** 0 FT	50 FT	100 FT	150 FT	200 FT
* 30 yr	19,345,662	18,895,578	18,848,274	18,900,310	18,750,990
50 yr	23,046,212	22,609,766	22,636,042	22,605,814	22,432,944
70 yr	23,848,500	23,923,700	23,826,588	23,970,604	23,710,012
100 yr	24,015,170	24,026,208	24,008,318	24,037,688	23,996,740

b) CHANGE IN AVERAGE HARVESTABLE VOLUME

	** 0 FT	50 FT	100 FT	150 FT	200 FT
* 30 yr	12,959,281	12,618,808	12,640,919	12,637,549	12,506,033
50 yr	12,368,536	12,055,621	12,059,902	12,013,713	11,927,325
70 yr	11,528,570	11,532,907	11,416,853	11,521,129	11,422,457
100 yr	10,948,496	10,947,168	10,870,771	10,958,235	10,966,565

c) COST (PNW$)

	* 30 yr	50 yr	70 yr	100 yr
** 0 FT	0	0	0	0
50 FT	450,084	436,446	-75,200	-11,038
100 FT	497,388	410,170	21,912	6,852
150 FT	445,352	440,398	-122,104	-22,518
200 FT	594,672	613,268	138,488	18,430

* :the time horizon
** : buffer size

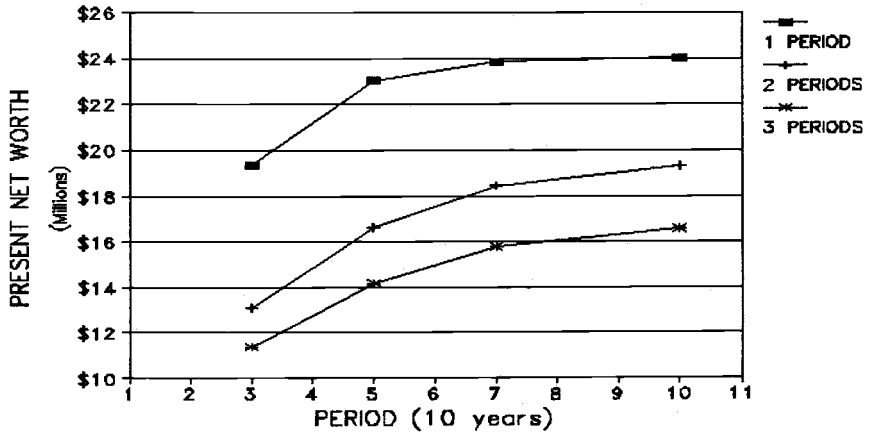
Change In Adjacency Lag Periods

The lag-scenario utilizes restrictions on the adjacency lag periods. No riparian management area is considered. The alternative number of adjacency lag periods is 2 and 3. Applying 2 periods for the adjacency lag, there is at least 1 period (10 years) during which no harvest occurs among adjacent harvest units, while in the other alternative period, at least for 2 periods (20 years) harvest has to be prohibited among adjacent units, after one unit of adjacent units is harvested.

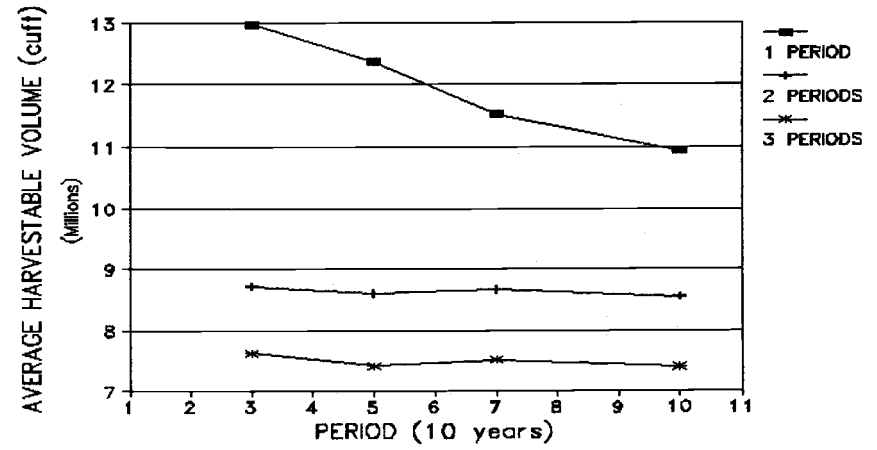
Figure 4.7-a illustrates the changing patterns of the objective value of each solution. The objective value was vastly reduced when using 2 or 3 periods for the adjacency lag. While the objective value for the 1-period adjacency lag increases from $19 million for the 3-period time horizon to $24 million for the 10-period time horizon, it increases from $13 million to $19 million for the 2-period adjacency lag and from $11 million to $16 million for the 3-period adjacency lag.

Figure 4.7-b depicts changing patterns of the average harvestable volume. Although the average harvestable volume for other scenarios in Figures 4.4, 4.5, and 4.6 decreases as the time horizon increases, the average harvestable volume for the solution with the 2- and the 3-period adjacency lag does not change much as the time horizon

a)



b)



c)

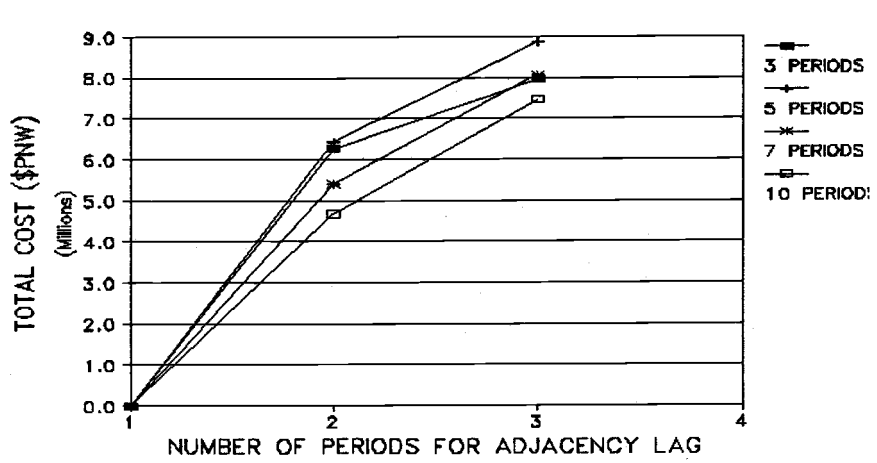


Figure 4.7. Results from the lag-scenario
a) Changing patterns of the objective value
b) Changing patterns of the average harvestable volume
c) Changing patterns of the derived cost

increases. It can be noted that restriction on the adjacency lag plays a more restricted role in determining the harvestable volume than the harvest flow constraints.

The derived cost in the lag-scenario is much higher than those in the static- and the dynamic-scenarios (Figure 4.7-c). Using the 3-period time horizon, the derived cost was about $6.2 million and $8.0 million for the 2- and the 3-period adjacency lag problems (Table 4.6). For the 5-period time horizon, they were $6.4 million and $8.9 million for the 2- and the 3-period adjacency lag problems. The derived cost was $5.4 million and $8.1 million for the 2- and the 3-period adjacency lag problems in the 10-period time horizon. As can be seen in Figure 4.7-c, for the 10-period time horizon the cost was less than others. For the 5-period time horizon it was the largest.

Evaluating Marginal Cost

Using the derived cost from each scenario, the marginal cost of implementing spatial restrictions on the example forest is estimated. The marginal cost is estimated by using the following simple regression model,

$$Y = a \cdot X \qquad (14)$$

where X is the buffer size for both the first and the second scenario, and the period for adjacency lag for the last scenario, and Y is the total cost ($PNW).

Table 4.6. Summary of results from the lag-scenario

a) CHANGE IN OBJECTIVE VALUE

	**1 period	2 periods	3 periods
* 30 yr	19,345,662	13,093,606	11,375,829
50 yr	23,046,212	16,613,337	14,174,449
70 yr	23,848,500	18,444,110	15,773,327
100 yr	24,015,170	19,333,216	16,560,778

b) CHANGE IN AVERAGE HARVESTABLE VOLUME

	**1 period	2 periods	3 periods
* 30 yr	12,959,281	8,719,321	7,625,953
50 yr	12,368,536	8,607,086	7,411,273
70 yr	11,528,570	8,666,422	7,522,089
100 yr	10,948,496	8,552,016	7,402,103

c) COST (PNW$)

	* 30 yr	50 yr	70 yr	100 yr
**1 period	0	0	0	0
2 periods	6,252,056	6,432,875	5,404,390	4,681,954
3 periods	7,969,833	8,871,763	8,075,173	7,454,392

* :the time horizon
** : adjacency lag periods

Taking the first derivative of Y with respect to X in equation (14), the marginal cost is calculated by,

$$\frac{d\,Y}{d\,X} = a \qquad (15)$$

That is, the coefficient "a" represents the marginal cost in the above regression model.

Table 4.7 shows the marginal cost, "a", of each problem and its ρ-value, where the ρ-value is defined as the probability that the marginal cost is not different for the respective methods and constraints. For the 3-period time horizon, the marginal cost for the first scenario by SSMART is lower, $14,954.59, than that by the approach without using SSMART. In the second scenario, the marginal cost is about 23% of the one in the first scenario, while the marginal cost in the third scenario is about 174 times more than that of the first scenario. The same relationship of the marginal cost among different scenarios can be found for the 5-, the 7-, and the 10-period time horizon. The third scenario provided much higher marginal cost than others, followed by the first scenario, and then the second scenario.

Judging from the ρ-value in Table 4.7, the marginal cost for the 7- and the 10-period time horizon in the second scenario can be regarded as zero. No change in the objective value is expected as the buffer size increases or decreases for these problems. For other problems, it is expected that the larger the buffer size, the less the

Table 4.7. Marginal cost for different scenarios

	* STATIC CHANGE IN BUFFER SIZE				**DYNAMIC		***ADJACENCY	
	SSMART		NO OPT.		SSMART		SSMART	
	M-Cost/ft	ρ-value	M-Cost/ft	ρ-value	M-Cost/ft	ρ-value	M-Cost/pd	ρ-value
3 periods	14,954.59	0.000	15,211.09	0.000	3,439.73	0.017	2,600,972	0.000
5 periods	17,951.73	0.000	17,747.38	0.000	3,354.03	0.012	2,820,074	0.000
7 periods	17,860.75	0.000	18,285.81	0.000	104.17	0.819	2,502,449	0.000
10 periods	17,375.44	0.000	18,245.70	0.000	5.88	0.935	2,266,219	0.000

M-Cost/ft:marginal cost per foot from the stream
M-Cost/pd:marginal cost per adjacency lag period
SSMART:SSMART applied to all constrained problems
NO OPT:no use of SSMART to constrained problems
* :the static-scenario
** :the dynamic-scenario
***:the lag-scenario

objective value and that the more the adjacency lag period, the less the objective value.

CONCLUSIONS

The objective of this paper was to estimate the cost of implementing spatial restrictions on the riparian zone of the example forest from the Alsea Ranger District, Siuslaw National Forest in Oregon. In order to analyze the effect of alternative regulations on riparian zone planning, three different scenarios were proposed.

The first scenario, the static-scenario, was modeled using the static change in the riparian management zone size, or buffer size. Following the Oregon Forest Practice Rules (1988), no harvest occurs within the riparian management area. With or without using the heuristic optimal harvest scheduling system, the objective value of the solution was almost the same, resulting in similar derived cost of implementing spatial restrictions. The marginal cost per foot of the riparian management area length changed from $1,495.59 to $17,951.73 for the use of the heuristic technique, SSMART, on all constrained problems, while it changed from $15,211.09 to $18,285.81 for the approach without the use of the SSMART on the constrained problems. Although the approach without the use of SSMART on the constrained problems provided values close to those estimated using SSMART, the corresponding harvest schedule was inferior to that developed by SSMART in terms of the actual flow fluctuation. The solution without the

use of SSMART tended to indicate that the larger the size of the riparian management area, the more the actual flow fluctuation of the harvest flow level. This was not the case for the solution using SSMART.

Allowing harvest to occur within the riparian management area was the second scenario (the dynamic-scenario). It was modeled by treating segments of the riparian management area as another harvest unit with additional adjacency constraints among segments and the original harvest units. Because of the availability of harvest units, the solution did not differ much in the objective value from that without any restrictions. The marginal cost per foot of buffer width from the stream was almost zero for the 7- and the 10-period time horizon and $3,439.73 and $3,354.03 for the 3- and the 5-period time horizon, respectively.

Finally a third scenario, the lag-scenario, was proposed. The objective value of the 2- and the 3-period adjacency lag problems was much lower than the one using the 1-period adjacency lag, resulting in high cost. The marginal cost per adjacency lag period was also high, changing from $2,266,219 to $2,820,074. While the average harvestable volume for the static- and the dynamic-scenarios decreases as the time horizon increases, it stays almost at the same level over different time horizons for the third scenario with the 2- and the 3-period adjacency lag.

The heuristic harvest scheduling system, SSMART, can be of use in solving the spatially constrained area-based harvest scheduling problem with even-flow constraints to provide a "good" solution. To estimate the cost of implementing spatial restrictions, the long-term harvest scheduling problem is solved. For these kinds of problems the use of the exact solution techniques such as the branch-and-bound algorithm in integer programming, could be impractical in terms of both computational time and dimensionality. Despite the lack of guaranteed optimality, SSMART can be one of several alternative techniques to create a "good" solution for the long-term scheduling problem with riparian constraints so that estimates of cost can be provided with ease for alternative policy selection.

LITERATURE CITED

Arney, J.D. 1985. A modeling strategy for the growth projection of managed stands. Can. J. For. Res., 15:511-518.

Barker, B.R. 1989. Utilizing scheduling and network analysis program (SNAP) as a forest plan implementation tool on the Siuslaw National Forest. M.S. thesis, College of Forestry, Oregon State Univ., Corvallis, OR. 76p.

Elmore, W. and R.L. Beschta. 1987. Riparian areas: perceptions in management. Rangelands Vol.9. No.6:260-265.

Johnson, K.N. and S.A. Crim. 1986. FORPLAN version 1: structures and options guide. USDA Forest Service Land Management Planning Systems Section, Washington, D.C. 276p.

Johnson, K.N., T.W. Stuart and S.A. Crim. 1986. FORPLAN version 2: an overview. USDA Forest Service Land Management Planning Systems Section, Washington, D.C. 98p.

Johnson, K.N. and T.W. Stuart. 1987. FORPLAN version 2: Mathematical programmer's guide. USDA For. Serv., Land Management Planning Systems Section, Washington, D.C. 158p.

Kirby, M.W., P. Wong, W.A. Hager and M.E. Huddleston. 1980. Guide to the Integrated Resource Planning Model. USDA

Forest Service, Berkeley, CA. 212p.

Meneghin, B.J., M.W. Kirby, and J.G. Jones. 1988. An algorithm for writing adjacency constraints efficiently in linear programming models. The 1988 Symp. on Systems Analysis in Forest Resources. USDA For. Serv. Gen. Tech. Rep. RM-161:46-53. Rocky Mountain Forest and Range Exp. Stn., Fort Collins, CO.

Navon, D.I. 1971. Timber RAM: a long range planning method for commercial timber lands under multiple use management. USDA Forest Service Res. Pap. PSW-70. 22p.

Nelson, J.D., J.D. Brodie and J. Sessions. 1990. Integrating short-term, area-based logging plans with long-term harvest schedules. For. Sci. Monograph. In journal review.

O'Hara, A.J., B.H. Faaland, and B.B. Bare. 1989. Spatially constrained timber harvest scheduling. Can. J. For. Res. 19:715-724.

Olsen, E.D., D.S. Keough and D.K. Lacourse. 1987. Economic impact of proposed Oregon Forest Practices Rules on industrial forest lands in the Oregon coast range: a case study. Forest Research Laboratory. Oregon State University. Corvallis. Research Bulletin 61. 15p.

Sessions, J. and J.B. Sessions. 1988. SNAP - a scheduling and network analysis program for tactical harvest planning. In Proceedings of International Mountain Logging and Pacific Northwest Skyline Symp., Dec. 12-16

1988. Oregon State Univ., Corvallis, OR. pp.71-75.

Torres R., J.M. and J.D. Brodie. 1990. Adjacency constraints in harvest scheduling: an aggregation heuristic. Can. J. For. Res. In press.

Torres R., J.M., J.D. Brodie, and J. Sessions. 1990. The use of relaxation to solve the habitat dispersion problem. In journal review.

Yoshimoto, A. and J.D. Brodie. 1990. Comparative efficiency of algorithms to generate adjacency constraints. In journal review.

Yoshimoto, A., J.D. Brodie, and J.M. Torres R. 1990. A comparison of approaches for integrating spatial concerns into harvest scheduling. In journal review.

Chapter 5

CONCLUSIONS

This dissertation has presented the use of a heuristic algorithm called ROHO-PATH (Random Ordering Heuristic Optimization with Projection Alternative TecHnique) to solve a spatially constrained area-based harvest scheduling problem with even-flow constraints. The PATH algorithm was originally introduced to solve a stand level optimization problem, which is equivalent to an unconstrained optimization problem. The main advantage of using the PATH algorithm is that the multiperiod problem is partitioned into a subproblem period by period, resulting in a vast reduction of the computational burden. When solving this harvest scheduling problem by exact solution techniques, the number of decision variables and constraints increases dramatically as the time horizon increases. Thus, even today's computer technology may fail in solving the problem using commercial optimization software, e.g., LINGO/386. This is not the case for the PATH algorithm. Instead of solving the problem over the total time horizon simultaneously, the PATH algorithm solves the problem at each period iteratively, leading to reduction of solution time and dimensionality.

Since our problem is constrained by even-flow and adjacency constraints, the PATH algorithm and problem

specification must be modified to deal with these constraints and multi-stand harvest scheduling concerns. In the proposed technique, the problem was partitioned into a harvest flow minimization problem and a present net worth maximization over the time horizon. Respecification of the objective in the partitioned problem is necessary since even-flow constraints are most likely violated, and these infeasibilities are to be minimized. Taking into account the dynamics of forest growth, feasibility of a solution in the partitioned problem was expanded to both the current and the following periods.

Since the respecified problem is not exactly the same as the original problem, in which the objective is to maximize the present net worth of the sum of returns from harvest units with even-flow and adjacency constraints, the modified random search technique, ROHO, is a key factor in obtaining a "good" integer solution by bridging these two problems.

The advantage of the ROHO algorithm over other random search techniques is that ROHO utilizes the stochastic nature of the random number to create the ordered sequence from which harvest units are selected for a solution in order, while other techniques use the random number directly to determine if a harvest unit is selected. Thus, some viable candidates could be missed. Using ROHO, a set of feasible solutions is created by changing the ordering in

the sequence.

Incorporating the ROHO-PATH algorithm, the Scheduling System of Management Alternatives foR Timber-harvest (SSMART) was developed. SSMART can handle 500 harvest units with a 20-period time horizon problem as the largest problem. In comparison to the performance of LINGO/386 and RELAX, SSMART appears to yield a "good" feasible solution for all period problems with a reasonable computational time and fairly small flow fluctuation.

The cost evaluation of spatial restrictions on riparian zone planning was attempted for a forest on the Green River Subbasin on the Alsea Ranger District, Siuslaw National Forest in Oregon. Without use of any optimization or heuristic techniques, the cost evaluation could be done well. However, the derived solution tends to have larger harvest flow fluctuation as the conditions become more restricted. SSMART, on the other hand, provided a solution with consistently small flow fluctuation.

A wide variety of the applications with the ROHO-PATH algorithm can be implemented in spatially constrained harvest scheduling. Since the exact solution techniques are costly and fail in providing any solution for problems that are inherently infeasible, heuristic techniques will probably replace them for harvest scheduling problems requiring a high degree of spatial resolution with many stands and periods. The techniques could be modified in

future research to incorporate road network optimization or questions of forest fragmentation.

BIBLIOGRAPHY

Arney, J.D. 1985. A modeling strategy for the growth projection of managed stands. Can. J. For. Res. 15:511-518.

Barker, B.R. 1989. Utilizing scheduling and network analysis program (SNAP) as a forest plan implementation tool on the Siuslaw National Forest. M.S. thesis, College of Forestry, Oregon State Univ., Corvallis, OR. 76p.

Bazaraa, M.S. and J.J. Jarvis. 1977. Linear programming and network flows. John Wiley & Sons Inc., New York. 565p.

Berck, P. and T. Bible. 1984. Solving and interpreting large-scale harvest scheduling problems by duality and decomposition. For. Sci. 30:173-182.

Brooke, S.H. 1958. A discussion of random methods for seeking maxima. Opns. Res. 6:244-251.

Bullard, H.S., H.D. Sherali, and W.D. Klemperer. 1985. Estimating optimal thinning and rotation for mixed-species timber stands using a random search algorithm. For. Sci. 31:303-315.

Conley, W. 1980. Computer optimization techniques. Petrocelli Books Inc., New York. 266p.

Cunningham, K. and L. Schrage. 1989. The LINGO modeling language. LINDO Systems Inc., Chicago, IL. 93p.

Dakin, R.J. 1965. A tree search algorithm for mixed integer programming problems. Computer J. 8:250-255.

Dantzig, G.B. 1951. Maximization of a linear function of variables subject to linear inequalities. Chap. 21 in: Activity Analysis of Production and Allocation, (T.C. Koopmans, ed.) Cowles Commission Monograph No. 13, Wiley, New York.

Elmore, W. and R.L. Beschta. 1987. Riparian areas: perceptions in management. Rangelands Vol.9. No.6:260-265.

Gavish, B. and H. Pirkul. 1985a. Zero-one integer programs with few constraints - lower bounding theory. Europ. J. Opns. Res. 21:213-224.

Gavish, B. and H. Pirkul. 1985b. Efficient algorithms for solving multiconstraint zero-one knapsack problems to optimality. Math. Prog. 31:78-105.

Geoffrion, A.M. 1974. Lagrangean relaxation for integer programming, in:M.L. Balinski, ed., Math. Prog. Study No.2 (North-Holland, Amsterdam):82-114.

Ghandforoush, P. and B.J. Greber. 1986. Solving allocation and scheduling problems inherent in forest resource management using mixed-integer programming. Comput. & Opns. Res. 13:551-562.

Glover, F. 1975. Surrogate constraint duality in mathematical programming. Opns. Res. 23:434-453

Greenberg, H.J. and W.P. Pierskalla. 1970. Surrogate mathematical programming. Opns. Res. 18:924-939.

Gross, T.E. and D.P. Dykstra. 1988. Harvest Scheduling with nonadjacency constraints. In Proceedings, Society of American Foresters National Convention, Oct. 16-19, 1988. Wash. D.C.:310-315.

Gross, T.E. 1989. Use of graph theory to analyze constraints on the Juxtaposition of timber stands. M.S. thesis. School of Forestry, Northern Arizona University, Flagstaff, AR. 153p.

Held, M., P. Wolfe, and H.P. Crowder. 1974. Validation of subgradient optimization. Math. Prog. 6:62-88.

Hoganson, H.M. and D.W. Rose. 1984. A simulation approach for optimal timber management scheduling. For. Sci. 30:220-238.

Hokans, R.H. 1983. Evaluating spatial feasibility of harvest schedules with simulated stand-selection decisions. J. For., 81:601-603, 613.

Johnson, K.N. and S.A. Crim. 1986. FORPLAN version 1: structures and options guide. USDA Forest Service Land Management Planning Systems Section, Washington, D.C. 276p.

Johnson, K.N., T.W. Stuart and S.A. Crim. 1986. FORPLAN version 2: an overview. USDA Forest Service Land Management Planning Systems Section, Washington, D.C. 98p.

Johnson, K.N. and T.W. Stuart. 1987. FORPLAN version 2: Mathematical programmer's guide. USDA For. Serv., Land Management Planning Systems Section, Washington, DC. 158p.

Kirby, M.W., P. Wong, W.A. Hager, and M.E. Huddleston. 1980. A guide to the integrated resources planning model. USDA For. Serv., Berkeley, CA. 211p.

Land, A.H. and A.G. Doig. 1960. An automatic method for solving discrete programming problems. Econometrica 28:497-520.

May, K.O. 1965. The origin of the four-color conjecture. Isis, 56:346-348.

Mealey, S.P., J.F. Lipscomb, and K.N. Johnson. 1982. Solving the habitat dispersion problem in forest planning. Trans. N. Amer. Wildl. Natur. Resour. Conf., 47:142-153.

Meneghin, B.J., M.W. Kirby, and J.G. Jones. 1988. An algorithm for writing adjacency constraints efficiently in linear programming models. The 1988 Symp. on Systems Analysis in Forest Resources. USDA For. Serv. Gen. Tech. Rep. RM-161:46-53. Rocky Mountain Forest and Range Exp. Stn., Fort Collins, CO.

Navon, D. 1971. Timber RAM - a long-range planning method for commercial timber lands under multiple use management. USDA For. Serv. Res. Paper PSW-70. Pacific Southwest Forest and Range Exp. Stn., Berkeley, CA. 22p.

Nelson, J., J.D. Brodie, and J. Sessions. 1988. Integrating short term spatially feasible harvest plans with long term harvest schedules using Monte-Carlo integer programming and linear programming. The 1988 Symp. on System Analysis in Forest Resources, March 29 - April 1 1988. Gen. Tech. Rep. Rm-161:224-229, Rocky Mtn. For. & Ran. Exp. Stn., Ft. Collins, Colorado.

Nelson, J., J.D. Brodie, and J. Sessions. 1990. Integrating short-term, area-based logging plans with long-term harvest schedules. For. Sci. Monograph. In journal review.

Nelson, J. 1988. Integration of short-term spatially feasible harvesting plans with long-term harvest schedules using Monte-Carlo integer programming and linear programming. Ph.D. Dissertation, Dept. of Forest Management, Oregon State Univ., Corvallis, OR. 168p.

Nemhauser, G.L. and L.A. Wolsey. 1988. Integer and combinatorial optimization. John Wiley & Sons Inc. New York. 763p.

O'Hara, A.J., B.H. Faaland, and B.B. Bare. 1989. Spatially constrained timber harvest scheduling. Can. J. For. Res. 19:715-724.

Olsen, E.D., D.S. Keough and D.K. Lacourse. 1987. Economic impact of proposed Oregon Forest Practices Rules on industrial forest lands in the Oregon coast range: a case study. Forest Research Laboratory. Oregon State University. Corvallis. Research Bulletin 61. 15p.

Paredes V., G.L. and J.D. Brodie. 1987. Efficient specification and solution of the even-aged rotation and thinning problem. For. Sci. 33:14-29.

Schrage, L. 1979. A more portable Fortran random number generator. ACM Trans. Math. Softw. 5:132-138.

Sessions, J. 1987. A heuristic algorithm for the solution of the variable and fixed cost transportation problem. In Proc: The 1985 Symp. on System Analysis in Forest Resources. Univ. of Georgia, Athens, GA. pp.324-336.

Sessions, J. and J.B. Sessions. 1988. SNAP - a scheduling and network analysis program for tactical harvest planning. In Proceedings of International Mountain Logging and Pacific Northwest Skyline Symp., Dec. 12-16 1988. Oregon State Univ., Corvallis, OR. pp.71-75.

Thompson, E.F., B.G. Halterman, T.J. Lyon, and R.L. Miller. 1973. Integrating timber and wildlife management planning. Forestry Chron., Dec. 1973:247-250.

Torres R., J.M. and J.D. Brodie. 1990. Adjacency constraints in harvest scheduling: an aggregation heuristic. Can. J. For. Res. In press.

Torres R., J.M. and J.D. Brodie. 1989. Manual to run program RELAX. Unpublished manuscript.

Torres R., J.M., J.D. Brodie, and J. Sessions. 1990a. Solution to the area-based harvest scheduling problem through Lagrangean relaxation. In journal review.

Torres R., J.M., J.D. Brodie, and J. Sessions. 1990b. The use of relaxation to solve the habitat dispersion problem. In journal review.

Williams, H.P. 1974. Experiments in the formulation of integer programming problems. In: M.L. Balinski, ed., Math. Prog. Study No.2 (Approaches to Integer Programming, North-Holland, Amsterdam):82-114.

Winston, W.L. 1987. Operations research: applications and algorithms. Duxbury Press, Boston. 1025p.

Yoshimoto, A., G.L. Paredes V., and J.D. Brodie. 1988. Efficient optimization of an individual tree growth model. The 1988 Symp. on Systems Analysis in Forest Resources. USDA For. Serv. Gen. Tech. Rep. RM-161:154-162. Rocky Mountain Forest and Range Exp. Stn., Fort Collins, CO.

Yoshimoto, A., R.G. Haight, and J.D. Brodie. 1990. A comparison of the pattern search algorithm and the modified PATH algorithm for optimizing an individual tree model. For. Sci. In press.

Yoshimoto, A. and J.D. Brodie. 1990. Comparative efficiency of algorithms to generate adjacency constraints. In journal review.

Yoshimoto, A., J.D. Brodie, and J.M. Torres R. 1990. A comparison of approaches for integrating spatial concerns into harvest scheduling. In journal review.

Zanakis, S.H. and J.R. Evans. 1981. Heuristic "optimization":why, when, and how to use it. Interfaces 11:84-91.