Development of a Genetic Algorithm Model for a Multiple-Objective Forest Harvesting and Zonation Problem

by

Tevfik Ziya Kuloglu

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Forest Biology and Management

Department of Renewable Resources

University of Alberta

Abstract

Most of the forested lands in North America are managed for multiple objectives and forest management plans are designed in a manner to obtain or maintain sustainable forest management certification. One strategy for achieving some of these goals is forest zonation, which allows for different intensities of forest management (including reserve areas) in different zones of the forest. The focus of this thesis is the development of a spatially explicit forest estate model which simultaneously allocates forest land to different management intensity zones and harvest periods with the goal of satisfying multiple management objectives. The model was initially developed using mixed integer goal programming. This helped to identify a basic model structure which was used to guide the development of a genetic algorithm-based implementation. The development of the model, particularly the setting of goal weights to describe the decision maker's preferences is described in detail.

Dedicated to my parents *Figen Kuloğlu* and *Mustafa Kuloğlu*

Acknowledgements

I would like to express my deepest gratitude to my advisor, Dr. Glen Armstrong, for his excellent guidance, caring, patience and providing me with an excellent atmosphere for doing research. Thank you to Dr. Martin Luckert and Dr. Scott Jeffrey for taking the time to review the document and participating as members of my examination committee. I would like to thank the Turkish Government for providing the opportunity and the financial support for my study. I am grateful to Value Chain Optimization Network for supporting my research and giving me valuable opportunities to meet scientists and members of the forestry sector. Alberta-Pacific Forest Industries Inc. provided the forest inventory data used here. I would also like to acknowledge the academic and technical support of University of Alberta, Department of Renewable Resources and its staff. Last but not the least, I would like to thank my family and my friends for long distance support and encouraging me throughout my study and my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Forest management plans are often designed to provide a wide range of economic, social, and ecological benefits. Drawing upon ideas related to sustainable forest management and forest zonation, this study develops mathematical programming and heuristic optimization models to solve a spatially explicit harvest planning problem that has multiple objectives related to sustainable forest management. This model is developed in several stages. First I formulate a mixed integer programming model that finds the forest management schedule that maximizes net present value from a small forested area while satisfying periodic harvest volume and adjacency constraints. This is model is then reforumulated as a mixed integer goal programming model by recasting the objective function and volume flow constraints as goal constraints, and changing the objective function to minimize weighted deviations from the goals. Finally the model is reformulated as a genetic algorithm. This allows me to include goals that would violate the assumptions of mixed integer programming, such as spatial configuration of reserve areas. This introductory chapter will briefly review the concepts of sustainable forest management and the history of modelling tools that have been used to help develop sustainable forest management plans.

1.1 Sustainable Forest Management

The idea of *Sustainable Development* gained prominence after the release of the Brundtland Commission Report (Report of the World Commission on Environment and Development: *Our Common Future*) in 1987 (Brundtland, 1987). In the report, sustainable development was defined as "…development that meets the needs of the present without compromising the ability of future generations to meet their own needs." The concept of sustainable development was applied to forest management at the United Nations Conference on Environment and Development (UNCED, 1992) in Rio de Janeiro, resulting in the "Forest Principles". (UNCED, 1992) applied the concepts of sustainable development to forest management. A main goal of the Montréal Process (1994) was to develop internationally agreed-upon criteria and indicators for sustainable forest management (SFM). The twelve member countries of the Montréal Process, which include Canada, represent 90% of the area of the world's temperate and boreal forests and 60% of the world's total forest area. In Canada, the criteria and indicators generated by the Montréal Process have been adopted by the Canadian Council of Forest Ministers' (CCFM) as its framework for forest management. This framework was first released in 1995 (CCFM, 1995), and a revised framework was released in 2003 (CCFM, 2003). The six criteria of sustainable forest management in the revised framework are

1. biological diversity,
2. ecosystem condition and productivity,
3. soil and water,
4. role in global ecological cycles,
5. economic and social benefits, and
6. society's responsibility.

Sustainable forest management requires consideration of many values (ecological, economic, and social) that are affected by forest management decisions.

1.1.1 Forest Planning Models

The aim of forest planning is to provide support for forestry decision making. Each stand in a forest has several different treatment schedules that are possible alternatives for it, and so each constitutes its own forest planning problem. Forest planning can be strategic, tactical or operational in nature. The main idea of strategic planning is to define what is desired from a forest. Tactical planning identifies how objectives for forest are obtained. Operational planning provides detailed recommendations for each stand of a forest (Kangas et al., 2008). Forest management plans are developed to focus on maximizing or minimizing objectives which may include environmental effects. The strategic level of forest management focuses on long-term goals (Bettinger and Sessions, 2003).

The large number of variables and constraints of many forest planning problems can make them difficult to solve. Addressing adjacency constraints is important for spatial harvest scheduling because it limits the harvesting opening size, which may be required by forest law or policy. One of the critical components of spatial forest management modeling is habitat fragmentation, which refers to the loss or reduction of habitat and the lack of connection between habitats and natural sources. Forest management planning problems should be examined in terms of forest values and structures, wildlife habitat, biodiversity, recreation and other purposes (Shan et al., 2009). Planning problems in forestry must be considered in terms of environmental issues as well as restrictions and goals. Optimization models and methods have been applied to solving forest planning problems. The diversity of decision problems and size of planning tasks have greatly increased. More information and more restrictions have led to larger models and higher complexity (Ronnqvist, 2003).

Sustained yield of timber harvest volume as the basis for forest regulation is probably the original timber sustainability criterion. Sustained yield has been a goal of forest management for a very long time. There have been several formula based methods that consider timber inventory and/or growth rates. Computerized models to determine sustainable timber harvest levels have been in use since the 1960's. Early forest planning methods were based on regulated even-aged forests that had a steady-state structure. The harvest area can be calculated by dividing the total area by the selected rotation time. The growth is equal to the annual harvest, and it is the same for each year's harvest once the forest reaches the regulated stage. The idea is sustained yield management. Traditional forest planning methods are based on area control and volume control. The area control method regulates the harvest area, which, in theory, will lead to a sustained yield of timber harvest volume after one rotation.

The volume control technique uses forest growth and age class to decide the allowable harvest levels (Kangas et al., 2008). There are a number of volume control formula that have been used to determine sustainable timber harvest levels.

The Austrian formula (Eq. 1.1) assumes that the cut should be equal to the growth, plus or minus an adjustment factor. The adjustment factor considers the difference in volumes between desired and actual stand volume (Leuschner et al., 1990).

$$\text{Annual Allowable Cut} = \left(\frac{V_g - V_f}{a}\right) + I \tag{1.1}$$

where V_g is the current growing stock volume and V_f is the desired growing stock volume of a regulated forest. The arbitrary number of decades used for the growing stock adjustment period is represented as a.

The Von Mantel formula (Eq. 1.2) estimates allowable harvest levels for even-aged stands. The assumption of the Von Mantel formula is based on the inventory volume of regulated forest, growing stock of stand is constant with age.

$$\text{Annual Allowable Cut} = \left(\frac{2V_g}{R}\right) \tag{1.2}$$

where V_g represents the volume of standing growing stock and R is the rotation age.

The Hanzlik formula (Eq. 1.3) is another method that can be used to determine harvest volume which ensures that the supply of wood would meet the demands at the same time. The Hanzlik formulate is used to determine sustained annual yields while switching virgin forests to normal forests and can be applied at the stand or forest level for stable harvest volume (Bettinger et al., 2010).

$$\text{Annual Allowable Cut} = \left(\frac{V_m}{R}\right) + I \tag{1.3}$$

where V_m represents the volume of mature timber above rotation age. The years in rotation age can be represented as R and mean annual immature timber is I.

The next development in forest planning was the use of computer models to determine sustainable timber harvest levels. One of the first computerized forest planning models was based on the area-volume check method (ARVOL) (Chappelle, 1966). ARVOL determines the constant annual harvest level that cuts the entire forest area exactly one over a rotation period. The Short Run Allowable Cut method (SORAC) calculates allowable cut using either area or volume regulation at the beginning of each planning period within a rotation. This program enables a timber management planner to trace future allowable cut over time, assuming period recalculation of the allowable cut (Chappelle and Sassaman, 1968). The Simulating Intensively Managed Allowable Cut program (SIMAC) is a computerized forest simulation model that estimates the sustainable allowable cut under intensive management regimes including thinning, mortality salvage, and genetic improvement. It computes a harvest rate based on present inventory and projected growth (Sassaman et al., 1972). It also uses a binary search method to find the maximum sustainable harvest level. These early computer based models (ARVOL, SORAC, and SIMAC) focussed on finding a maximum sustained yield of timber considering forest inventory and growth assumptions.

Beginning in the late 1960s, mathematical programming models of the forest planning began to appear. These new models allowed for the specification of alternative objective functions (*e.g* maximize timber harvest volume, minimize costs, or maximize net present value) subject to constraints on other outputs of interest. We begin to see multiple objectives for forest management be explicitly taken into account in forest planning models. One example of mathematical programming within forest planning systems includes the Timber Resource Allocation Method (RAM) (Navon, 1971). Timber RAM was an early linear programming model developed to address the question of biological sustainability of a harvest level on a forest-wide basis. Another example is Max-million (Clutter, 1968) which was developed for forest products firms and model had merits for timber harvest and activity scheduling.

The Multiple Use-Sustained Yield Calculation (MUSYC) technique (Johnson and Jones, 1979) was an attempt to improve Timber RAM and provided for the integration of other forest uses into timber planning. MUSYC then became a prototype for the development of FORPLAN (Johnson et al., 1986). SPECTRUM (Greer and Meneghin, 1991) is a graphical user interface version of FORPLAN. Both are highly flexible modeling programs that incorporate the strata-based information required for per acre yields as well as an innovative area based formulation required by recreation, wildlife, water and other environmental outputs needing area-wide yields. These areas, also called *zones*, could receive totally different sets of management actions depending on the zones (Von Gadow et al., 2000). Spatial Woodstock (Remsoft Inc, 2002) is a forest modeling software that deals with management planning problems such as inventory projections, long-term harvest schedules, land or silviculture investment queries, biodiversity and, wild-life habitat evaluations. These systems all take a representation of the forest planning problem and translate it into a mathematical programming matrix, send the matrix to a mathematical program solver, and then translate the output of the solver into outputs designed to be easily interpreted by a forest planner. Linear programming is the most widely used mathematical programming method in forest resources management.

Recently, there has been an interest in heuristic algorithms that potentially allow for more integer variables and a more flexible relationship than possible through linear programming and related mathematical programming techniques. Mathematical programming techniques, including traditional linear programming (e.g. Weintraub et al., 1995), dynamic programming (e.g. Hoganson and Borges, 1998), Monte Carlo integer programming (e.g. O'Hara et al., 1989), simulated annealing (e.g. Lockwood and Moore, 1993), tabu search (e.g. Murray and Church, 1995), threshold accepting (e.g. Bettinger and Sessions, 2003), and genetic algorithms (e.g. Mullen and Butler, 2000) have all been used for spatial forest planning. Patchworks (Spatial Planning Systems, 2009) is a commercially available program that uses heuristic methods to integrate operational-scale decision-making within a strategic-analysis environment and enables spatially explicit harvest allocations to be developed over large forest areas and long planning horizons.

Forest zoning is an alternative framework that allocates forest objectives into separate areas that each have independent management strategies. The aim of the TRIAD approach is to minimize the negative environmental effects on forest while maintaining timber production by dividing the forest into three or more broad land-use zones (Cote et al., 2010). Seymour and Hunter (1992) recommend this approach for solving conflict between environmental and industrial interests. Forest zoning limits activities in various zones by establishing values for a forest as well as objectives, criteria and indicators, and targets to be achieved (Boyland et al., 2004). The goal of TRIAD zonation approach is to allow timber harvest volumes to be maintained or possibly increased while simultaneously reducing the impact on the environment. A policy framework for a coexistence of plantation and reserve areas could be provided by a TRIAD zoning system. Anderson et al. (2012) used a forest management scheduling model to estimate how policy impacts the net present value of the optimal forest plan. Non-harvested areas were assigned in Anderson et al. (2012) as reserve area. This thesis aims to find a reserve value according to stand age to decide which areas will be allocated as reserve zones.

1.2 Planning Techniques: Mathematical Programming Models

The decision-making process in sustainable forest management planning is complicated because of the multitude of objectives that need to be considered including protection of ecosystems, the desire for economic development, wood supply and demand, and habitat preservation (Kaiser et al., 2011). The mathematical programming models examined here are linear programming and some of its extensions. A mathematical programming model finds the optimum value (maximum or minimum) of an *objective function* which is expressed as a linear function of a number of *decision variables*. In most forest planning mathematical programming models, a large number of the decision variables represent the amount of forested land of a particular type to be assigned to a particular management prescription, at a particular time. The model also includes a number of *constraints* which ensure that the solution to the model recognizes limitations on resource availability (*e.g.* forest land area) and policy goals (*e.g.* controls on periodic harvest volume).

1.2.1 Linear Programming

A linear programming model of a management problem must satisfy certain mathematical assumptions, including

1. *proportionality.* The contribution of each decision variable to the objective function is proportional to its activity level.
2. *additivity.* The effects of the decision variables are additive: there are no interactions among variables.

3. *divisibility.* The decision variables can take on any non-negative value, includeing fractions.

4. *certainty.* All model parameters (the objective function coefficents, constraint coefficients, and right-hand side constants) are known with certainty.

Together the proportionality and the addtivity assumptions mean that the objective function and constraints are linear functions. These are strict assumptions, and are unlikely to be satisfied perfectly in any forest management problems (Buongiorno and Gilless, 2003). However, these assumptions can be assumed to be a reasonable approximation of reality in many cases, and useful LP models have been developed.

1.2.2 Mixed Integer Programming

Integer programming techniques have been developed which require the decision variables to take on non-negative integer variables. Many models include both integer and continuous variables. These mixed models are usually called mixed integer programming (MIP) models (Bettinger et al., 2010). It is not hard to find forest manangement problems where the removal of the divisibility assumption would allow for a better representation of the problem. For example, many decisions might be best modeled as binary variables (*e.g.* harvest stand i in period j or not). In particular, many spatial forest management problems would lead to a treatment of a particular stand in a particular time period.

The techniques used to solve an MIP model (*e.g.* branch-and-bound or the cutting plane method) can be very time consuming. Because spatially explicit forest planning models typically contain many integer variables, it may be impractical to use MIP for realistically sized forest management models.

1.2.3 Goal Programming

A linear programming model must have exactly one objective function. However, there are methods to accumulate multiple objectives into an objective function to incorporate multiple goals of the decision maker. When the decision maker considers multiple objectives, goal programming methods would be useful (Bettinger et al., 2010). Goal programming is a form of linear programming that could be of value for multiple-use management considerations. Goal programming is a mathematical procedure to determine of a plan which proposes to minimize sum of the weighted deviations from goals.

1.3 Planning Techniques: Heuristic Methods

Some spatial forest management problems may be difficult to formulate as mixed integer programs because the nature of the problem does not satisfy the assumptions of MIP. In other cases, a reasonable formulation of the problem can be created, but the solution time required for a mixed integer program might be unacceptably long. In these cases, heuristic optimization methods are worth considering. Heuristic methods attempt to find a good solution to a problem at a reasonable cost in terms of computing power. There is no guarantee that the solution from an heuristic is optimal, but the hope is that it is near-optimal. Heuristic techniques are becoming popular for developing alternative forest plans that include spatial constraints. Heuristic methods can be used also for some management planning problems which have non-linear and/or complex constraints that must be included simultaneously with the scheduling of management activities (Bettinger et al., 2010).

There are a number of examples of different heuristic methods used for forest planning problems including Monte Carlo optimization (e.g. Nelson and Brodie, 1990), simulated annealing (e.g. Murray and Church, 1995), threshold accepting (e.g. Bettinger et al., 2003), tabu search (e.g. Bettinger et al., 1997) and genetic algorithms (e.g. Boston and Bettinger, 2002).

1.3.1 Monte Carlo Optimization

Monte Carlo optimization does not guarantee optimality, unlike mixed-integer programming; however, it is capable of generating feasible solutions to complex problems with a large number of integer variables. The Monte Carlo optimization technique consists of generating a random sample of number of feasible solutions and selecting the best one. The solution returned by a local search after exploring a neighborhood structure. In other words, a local search is applied repeatedly to obtain the local optima from the selected neighboring solution. Monte Carlo integer programming was used by Nelson and Brodie (1990) to solve a combined harvested scheduling and transportation planning problem with adjacency constraints. Monte Carlo integer programming model in Nelson and Brodie (1990) addressed several management goals in production harvest schedules with spatial constraints applied to examine the impact of cut-block sizes and adjacency delays.

Boston and Bettinger (2002) applied Monte Carlo integer programming model which randomly selects units for harvest, developing a schedule one planning period at a time, until the volume goal has been met for all periods, or until a user-specified number of solutions has been examined. Once all units have been selected, the objective function is calculated, and penalty values for deviations from the volume goals are applied. If the current solution is better than the best solution, the current solution replaces the best solution. Another Monte Carlo optimization model tested by Bettinger and Zhu (2006) which selects harvest schedule from a list of forest plans physically nearest the original harvest schedules, and the next best alternative for this plan, that does not result in a constraint violation with the originally selected forest plan, is chosen and forced into the solution.

1.3.2 Simulated Annealing

Simulated Annealing (SA) simulates the process of physical annealing, in which a metal is heated and then allowed to cool very slowly. SA is used to explore the solution space of a problem and allows for escapes from local optima in order to find a global optimum (Henderson et al., 2003). Simulated Annealing (SA) has been used in many forestry applications.

Lockwood and Moore (1993) developed a simulated annealing method for a harvest schedule problem because the size of the forest area is both far larger than could be handled by integer programming approaches. at the time.

The simulated annealing algorithm designed and illustrated by Boston and Bettinger (1999) to solve the harvest-scheduling problem is similar to Lockwood and Moore (1993). By adding more constraints to the problem tends to slow down the progress of the search. When a problem restricted with lots of constraints, feasible solutions are hard to find through a permutation operation, then penalty functions are needed for the heuristic to diversify its search (Crowe and Nelson, 2005).

1.3.3 Tabu Search

An alternative meta-heuristic approach to simulated annealing is tabu search. Unlike other heuristics, there is no random aspect within a general tabu search algorithm. Tabu search is designed to deal with local optimality in a more orderly fashion than simulated annealing.

Tabu search evaluates the potential changes to the plan and choose the best choice from the set. The short-term and long-term strategies help the process to search the solution space for better solutions by avoiding unproductive movement cycle. Defining the neighborhood, intensification, and diversification are the three main important elements of the tabu search (Murray and Church, 1995). Objective function or constraints are not require linearity, continuity or convexity. The tabu search method was applied and evaluated in Richards and Gunn (2003) to solve spatial forest planning and road access problem with spatial constraints.

1.3.4 Genetic Algorithms

Genetic algorithms (GA) were developed by Holland (1975) to formally study the phenomenon of adaptation as it occurs in nature and to develop ways in which the mechanisms of natural adaptation might be imported into computer systems (Mitchell, 1999). The search process of genetic algorithms is not based on neighborhood search as in simulated annealing or tabu search. The alternative solutions are called parent chromosomes, which are processed by crossing over (combining parts of two or more chromosomes) and by mutation (random change in one or several genes, or compartments).

The simple form of genetic algorithm involves three types of operators: selection, crossover, and mutation.

Selection selects chromosomes in the population for reproduction. The fitter the member, the more times it is likely to be selected to reproduce. After randomly generated population, fitness of each chromosome/individual in the population is calculated by using fitness function. Two parents are randomly selected from the population with the probabilities proportional to their fitness value. Each generation is the updated group of individual members which are replaced with initial population (Pukkala and Kurttila, 2005). Once the fitness has been calculated for all members of the population, members can be selected which are fit to become parents and place them in a mating pool (Shiffman et al., 2012). ***Crossover*** randomly chooses and exchanges the sequences so that two member create two offspring. For example, the strings 10000100 and 11111111 could be crossed after the third locus in each to produce the two offspring 10011111 and 11100100. ***Mutation*** randomly flips some of the bits in a gene. For example, the string 00000100 might be mutated in its second position to yield 01000100. Mutation can occur at each bit position in a string with some probability, usually very small (e.g., 0.001). The function will produce a numeric score to describe the fitness of a given member of population (Shiffman et al., 2012).

1.4 Conclusion

The objective of the thesis is to develop a method to solve a multiple-objective spatial explicit optimization problem using a gentic algorithm. In this thesis, I will focus on forest management planning from the perspective of spatial restrictions imposed by environmental concerns. The aim is to develop a model that will simultaneous zone the forest into harvest and reserve areas, and to select the period of harvest for the harvest areas. I explore genetic algorithms in this thesis because I believe they are under-represented in the forestry literature.

In chapter 2 of this thesis, I develop a representation of the forest planning problem as a mixed integer goal programming (MIGP) model. I use this model to help develop the structure of the fitness function to be used in the genetic algorithm model and to help set the target levels for goals.

In chapter 3, I present the genetic algorithm and use it to develop and to help set target levels for the different goals including net present value, harvest levels, adjacency violations, reserve quality, and a measure of the proximity of the stands that make up the reserve area. I explore mechanism for setting goal weights, and explore impacts of alternative parameters setting such as number of generations, population size, and mutation rates.

My hope is to develop an understanding of how genetic algorithm might work in a harvest scheduling problem and to create a prototype model that can be used as the basis for a real forest management problems.

The fourth and final chapter will present some overall conclusions and suggestions for further work.

Bibliography

Anderson, J. A., Armstrong, G. W., Luckert, M. K., Adamowicz, W. L., 2012. Optimal zoning of forested land considering the contribution of exotic plantations. Math. Comput. For. Nat.-Res. Sci. 4 (2), 92–104.

Bettinger, P., Boston, K., Siry, J., Grebner, D. L., 2010. Forest Management and Planning. Academic Press, San Diego, CA.

Bettinger, P., Johnson, S. J., Debora, L., Johnson, K. N., 2003. Spatial forest plan development with ecological and economic goals. Ecol. Model. 169 (2), 215–236.

Bettinger, P., Sessions, J., 2003. Spatial forest planning: To adopt, or not to adopt? J. Forest. 101 (2), 24–29.

Bettinger, P., Sessions, J., Boston, K., 1997. Using tabu search to schedule timber harvests subject to spatial wildlife goals for big game. Ecol. Model. 94 (2), 111–123.

Bettinger, P., Zhu, J., 2006. A new heuristic method for solving spatially constrained forest planning problems based on mitigation of infeasibilities radiating outward from a forced choice. Silva Fenn. 40(2), 315–333.

Boston, K., Bettinger, P., 1999. An analysis of monte carlo integer programming, simulated annealing, and tabu search heuristics for solving spatial harvest scheduling problems. Forest Sci. 45 (2), 292–301.

Boston, K., Bettinger, P., 2002. Combining tabu search and genetic algorithm heuristic techniques to solve spatial harvest scheduling problems. Forest Sci. 48 (1), 35–46.

Boyland, M., Nelson, J., Brodie, J Douglas Bunnell, F. L., 2004. Creating land allocation zones for forest management: A simulated annealing approach. Can. J. Forest Res. 34 (8), 1669–1682.

Brundtland, G. H., 1987. Report of the World Commission on environment and development: "our common future". United Nations.

Buongiorno, J., Gilless, J. K., 2003. Decision Methods for Forest Resource Management. Academic Press., New York.

CCFM, 1995. Defining sustainable forest management: A Canadian approach to criteria and indicators. Canadian Council of Forest Ministers, Ottawa. 22 p. Available at: http://www.ccfm.org/english/coreproducts-criteria_in.asp.

CCFM, 2003. Defining Sustainable forest management in Canada: Criteria and Indicators. Canadian Council of Forest Ministers, Ottawa. 21 p. Available at: http://www.ccfm.org/english/coreproducts-criteria_in.asp.

Chappelle, D. E., 1966. A computer program for calculating allowable cut using the area-volume check method. Research Note PNW-44, USDA Forest Service. Pacific Northwest Forest and Range Experiment Station, Portland, Oregon, USA.

Chappelle, D. E., Sassaman, R., 1968. A computer program for scheduling allowable cut using either area or volume regulation during sequential planning periods. Research Note. Portland. USDA Forest Service, Pacific Northwest Forest and Range Experiment Station.

Clutter, J. L., 1968. Max-million: a computerized forest management planning system. The University of Georgia. School of Forest Resources. Biometrics-Operations Research Section.

Cote, P., Tittler, R., Messier, C., Kneeshaw, D. D., Fall, A., Fortin, M.-J., 2010. Comparing different forest zoning options for landscape-scale management of the boreal forest: possible benefits of the triad. Forest Ecol. and Manag. 259 (3), 418–427.

Crowe, K. A., Nelson, J., 2005. An evaluation of the simulated annealing algorithm for solving the area-restricted harvest-scheduling model against optimal benchmarks. Can. J. Forest Res. 35 (10), 2500–2509.

Greer, K., Meneghin, B., 1991. Spectrum: An analytical tool for building natural resource management models. US Department of Agriculture Forest Service General Technical Report, 174–178.

Henderson, D., Jacobson, S. H., Johnson, A. W., 2003. The theory and practice of simulated annealing. In: Handbook of metaheuristics. Springer, pp. 287–319.

Hoganson, H. M., Borges, J. G., 1998. Using dynamic programming and overlapping subproblems to address adjacency in large harvest scheduling problems. Forest Sci. 44 (4), 526–538.

Holland, J. H., 1975. Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. University of Michigan Press.

Johnson, K., Jones, D., 1979. A users guide to multiple use sustained yield resource scheduling calculation (MUSYC). Washington, DC: USDA. Forest Service, Timber Management Staff. 242p.

Johnson, K. N., Stuart, T. W., Crim, S. A., 1986. Forplan version 2: An overview. Washington: USDA Forest Service, 80.

Kaiser, H. M., Messer, K. D., et al., 2011. Mathematical programming for agricultural, environmental and resource economics. John Wiley and Sons, Inc.

Kangas, A., Kangas, J., Kurttila, M., 2008. Decision support for forest management. Vol. 16. Joensuu, Finland. Springer.

Leuschner, W. A., et al., 1990. Forest regulation, harvest scheduling, and planning techniques. John Wiley & Sons, Inc.

Lockwood, C., Moore, T., 1993. Harvest scheduling with spatial constraints: a simulated annealing approach. Can. J. Forest Res. 23 (3), 468–478.

Mitchell, M., 1999. An introduction to genetic algorithms. Cambridge, Massachusetts London, England, Fifth printing 3.

Mullen, D. S., Butler, R. M., 2000. The design of a genetic algorithm based spatially constrained timber harvest scheduling model. United States Department of Agriculture Forest Service General Technical Report NC 1, 57–65.

Murray, A. T., Church, R. L., 1995. Heuristic solution approaches to operational forest planning problems. Oper. Res. Spectrum 17 (2), 193–203.

Navon, D. I., 1971. Timber RAM A long-range planning method for commercial timber lands under multiple-use management. U.S. Pacific Southwest Forest and Range Experiment Station. USDA Forest Service research paper PSW-70 Berkeley, Calif.

Nelson, J., Brodie, J. D., 1990. Comparison of a random search algorithm and mixed integer programming for solving area-based forest plans. Can. J. Forest Res. 20 (7), 934–942.

O'Hara, A. J., Faaland, B. H., Bare, B. B., 1989. Spatially constrained timber harvest scheduling. Can. J. Forest Res. 19 (6), 715–724.

Pukkala, T., Kurttila, M., 2005. Examining the performance of six heuristic optimisation techniques in different forest planning problems. Silva Fenn. 39 (1), 67–80.

Remsoft Inc, 2002. Spatial Woodstock users guide. Frederiction, Canada.

Richards, E. W., Gunn, E. A., 2003. Tabu search design for difficult forest management optimization problems. Can. J. Forest Res. 33 (6), 1126–1133.

Ronnqvist, M., 2003. Optimization in forestry. Math. Program. 97 (1-2), 267–284.

Sassaman, R. W., Holt, E., Bergsvik, K., 1972. User's manual for a computer program for simulating intensively managed allowable cut.

Seymour, R., Hunter, M. L., 1992. New forestry in eastern spruce-fir forests: principles and applications to Maine.

Shan, Y., Bettinger, P., Cieszewski, C., Li, R. T., 2009. Trends in spatial forest planning. Math. Comput. For. Nat.-Res. Sci. 1 (2), Pages–86.

Shiffman, D., Fry, S., Marsh, Z., 2012. The Nature of Code. Theoklesia, LLC.

Spatial Planning Systems, 2009. What is Patchworks? Available at: http://www.spatial.ca/products/index.html. [Accessed 2014-06-27].

UNCED, 1992. Agenda 21, The Rio Declaration on Environment and Development, the Statement of Forest Principles, the United Nations framework convention on climate change and the United Nations convention on biological diversity. Available at: http://www.un.org/geninfo/bp/enviro.html [Accessed 2014-04-21].

Von Gadow, K., Pukkala, T., Tomé, M., 2000. Sustainable forest management. Vol. 1. Springer.

Weintraub, A., Magendzo, A., Magendzo, A., Malchuk, D., Jones, G., Meacham, M., 1995. Heuristic procedures for solving mixed-integer harvest scheduling-transportation planning models. Can. J. Forest Res. 25 (10), 1618–1626.

Chapter 2

Mixed Integer Programming Model Formulation and Solution

2.1 Introduction

Mathematical programming models have been used for forest planning since the 1960's. Examples of mathematical programming forest planning systems include Max-million (Clutter, 1968), Timber RAM (Navon, 1971), MUSYC (Johnson and Jones, 1979), FORPLAN (Johnson et al., 1986), SPECTRUM (Greer and Meneghin, 1991), and Spatial Woodstock (Remsoft Inc, 2002). Linear programming is the most widely used mathematical programming method in forest resources management. One reason for this is the computational efficiency of linear programming. Another is the long history of linear programming based on forest management planning, which provides a strong foundation for the development of new software. It is also straightforward to incorporate growth and yield projections from practically any stand projection model.

Mathematical programming problems in forest management deal with determining optimal allocations of limited resources to reach given objectives by maximizing or minimizing a numeric function of a number of variables (Dykstra, 1984). The objective function in a linear programming model calculates a value for the main goal (e.g. total harvest volume or net present value) of a decision problem based on values assigned to decision variables. In forest planning models, decision variables are often used to represent the area of individual stands (or aggregates of stands) scheduled for a particular management action in a particular time period. Constraints restrict solutions to problems by limiting the choices for the decision variables (Bettinger et al., 2010).

Linear programming offers an optimal solution to a forest management problem that meets specific assumptions. The proportionality assumption means that each decision variable in the objective function must be a constant coefficient. The additivity assumption implies that the contribution of the objective function for any variable is independent of the other decision variables. Divisibility means that the variables can take on fractional values. A linear programming model assumes that the parameters in the model are known constants (Buongiorno and Gilless, 2003). A linear programming model is helpful for illustrating the economic relationships among rotation length, period for harvesting, and net present value of the forest (Steuer and Schuler, 1978). Linear programming does have limitations, however; not all decisions can be adequately represented by continuous variables. The divisibility assumption of linear programming limits variables because all decision variables must be able to take fractional levels. Many constraints and objective functions may be better represented by non-linear functions. Linear programming often suggests harvest of parts of stands or stand aggregates. In some forest management problems, particularly those for which spatial relationships need to be considered, it may be desirable to treat the entire stand or aggregate. I build several models, linear programming, mixed integer programming, and mixed integer goal programming to solve the harvest scheduling problem for a small forest.

2.2 Linear Programming Modeling Formulation

Harvest scheduling is the decision making process that specifies where to harvest, when to harvest and how much to harvest over a planning horizon to best achieve some predefined set objective. The harvest scheduling problem solved using linear programming in this study maximizes net present value. The linear programming optimization model in this study is formulated and solved using GUSEK (2013). GUSEK is a Microsoft Windows based integrated development environment for the GNU Linear Programming Kit (`http://www.gnu.org/software/glpk/`). See appendix A for the GUSEK code for the models developed here. This code is also available at the University of Alberta Education and Research Archive at `http://hdl.handle.net/10402/era.38715`.

The decision variables (Eq. 2.2), X_{ij}, represent the proportional area of stand i assigned for particular action in period j. The decision variables are restricted between 0 or 1 in this study. The variables in a linear program are a set of quantities that need to be determined in order to solve the problem. The decision variables represent the amount of a resource to use or the level of some activity. Total coniferous and deciduous harvest volume in each period of the planning horizon are represented in the model as accounting variables, $\mathrm{H}_{\mathrm{j}}^{cv}$ and $\mathrm{H}_{\mathrm{j}}^{dv}$.

The parameter a_i represents the area (ha) of stand i. The parameters vc_{ij} and vd_{ij} are the coniferous and deciduous harvest volumes (m^3/ha) that would be obtained from a harvested stand i in period j. The price parameters, pc and pd, (\$/m^3) are the prices paid for coniferous and deciduous harvests. I represents the discount rate, m_j is the midpoint of the harvest period. The present value of revenue for stand i harvested in period j is represented PR_{ij}^{c} for conifer trees (Eq. 2.3) and PR_{ij}^{d} for deciduous trees (Eq. 2.4). This value is calculated according to all revenue and costs accrued at the midpoint of a five-year planning period. The present value of cost parameter PC_{ij}^{hv} is the discounted harvest cost. h represents the cost of harvesting (\$/ha) in the equation Eq. 2.5. The total discounted haul cost (Eq. 2.6) is calculated by multiplying a stand's distance from the mill, haul cost, k (\$/m^3/km), area and harvest volume of each species. The distances between stand i and sawmill are represented in matrix as d_{is} and the distances between stand i and pulp mill is d_{ip}. The parameter shows the distances from centroid of each stand to each of two mills (softwood sawmill and hardwood pulp mill) which is calculated by using ArcGIS. The other parameter for net present value coefficient is calculated as total revenue minus total cost (Eq. 2.7).

The resource constraint that developed for this problem indicates that the sum of the proportional area assigned for harvesting over a given time period should not exceed the total area. In other words, the total proportion of the area harvested from the stand over the planning horizon should not exceed *1* (Eq. 2.8).

Equation 2.9 was used to calculate total coniferous and deciduous harvest volumes, $\mathrm{H}_{\mathrm{j}}^{cv}$ and $\mathrm{H}_{\mathrm{j}}^{dv}$, in each period of the planning horizon to satisfy accounting constraints.

Equation 2.10 represents even-flow policy constraints which ensure that coniferous and deciduous harvest volumes in each time period are equal to the previous period's volume.

The form of the objective function is as follows:

$$\max Z = \sum_{i=1}^{N} \sum_{j=1}^{T} PF_{ij} X_{ij} \tag{2.1}$$

where Z is the net present value for i = *1,...,N*.

The decision variables are represented as:

$$0 \leq X_{ij} \leq 1 \tag{2.2}$$

The present value of revenue for conifer harvest volume is calculated as in equation 2.3 :

$$PR^{c}_{ij} = \frac{vc_{ij} a_i pc}{(1+I)^{m_j}} \qquad \forall \quad \begin{array}{l} i = 1, ..., N; \\ j = 1, ..., T \end{array} \tag{2.3}$$

The present value of revenue for deciduous harvest volume is calculated as in equation 2.4 :

$$PR^{d}_{ij} = \frac{vd_{ij} a_i pd}{(1+I)^{m_j}} \qquad \forall \quad \begin{array}{l} i = 1, ..., N; \\ j = 1, ..., T \end{array} \tag{2.4}$$

The present value of harvesting cost, PC^{hv}_{ij}, and haul cost are calculated, PC^{hl}_{ij}, in equation 2.5 and 2.6 as follows:

$$PC^{hv}_{ij} = \frac{a_i h}{(1+I)^{m_j}} \qquad \forall \quad \begin{array}{l} i = 1, ..., N; \\ j = 1, ..., T \end{array} \tag{2.5}$$

$$PC^{hl}_{ij} = \frac{vc_{ij} a_i d_{is} k + vd_{ij} a_i d_{ip} k}{(1+I)^{m_j}} \qquad \forall \quad \begin{array}{l} i = 1, ..., N; \\ j = 1, ..., T \end{array} \tag{2.6}$$

$$PF = PR^{c}_{ij} + PR^{d}_{ij} - PC^{hv}_{ij} - PC^{hl}_{ij} \qquad \forall \quad \begin{array}{l} i = 1, ..., N; \\ j = 1, ..., T \end{array} \tag{2.7}$$

The resource constraint is calculated as follows:

$$\sum_{j=1}^{T} X_{ij} \leq 1 \qquad \forall \quad i = 1, ..., N \tag{2.8}$$

The accounting constraints are calculated as follows:

$$\begin{array}{l} \sum_{i=1}^{N} vc_{ij} X_{ij} a_j - H^{cv}_j = 0 \qquad \forall \quad j = 1, ..., T \\ \sum_{i=1}^{N} vd_{ij} X_{ij} a_j - H^{dv}_j = 0 \end{array} \tag{2.9}$$

The policy constraints are calculated as follows:

$$\begin{array}{l} (H^{cv}_j) - (H^{cv}_{j-1}) = 0 \qquad \forall \quad j = 2, ..., T \\ (H^{dv}_j) - (H^{dv}_{j-1}) = 0 \qquad \forall \quad j = 2, ..., T \end{array} \tag{2.10}$$

The non-negativity constraints represented in Eq. 2.11 ensure that all of the decision variables in the model take non-negative values.

$$X_{ij}, H^{cv}_j, H^{cv}_j \geq 0 \qquad \forall \quad \begin{array}{l} i = 1, ..., N; \\ j = 1, ..., T \end{array} \tag{2.11}$$

2.3 Mixed Integer Programming Modeling Formulation

When the values of some of the decision variables can take only integer values (either general integers or binary) instead of continuous values, mixed integer programming may be an appropriate technique to solve forest management problems. The optimization model in this study is formulated and solved by using a mixed integer programming (MIP) method that is programmed in GUSEK (2013). See appendix B for the GUSEK code. This code is also available at the University of Alberta Education and Research Archive at `http://hdl.handle.net/10402/`

`era.38715`. In this section, I developed a mixed integer formulation of the forest-planning problem that includes number of stands, *N*, and number of periods, *T*. The model includes adjacency constraint, and even-flow constraints are converted to sequential flow constraints that restrict harvest volume to within 5 % of the previous period's harvest volume. The even flow constraints are converted to the sequential flow constraints because when whole stand is harvested, the same amount of harvest volume in each time period is difficult to achieve.

The binary decision variables, X_{ij}, indicate whether or not stand i is harvested in period j. Unlike in the linear programming model, decision variables are binary in the mixed integer programming model. If $X_{ij}=1$, stand i is harvested in period j, if $X_{ij}=0$, stand i is not harvested in period j. In a linear programming model, $X_{ij}=1$ would represent the proportion of the area of stand i harvested in period j. Total coniferous and deciduous harvest volumes calculation in each period of the planning horizon are similar to the linear programming that are represented here as accounting variables, $\mathrm{H}_{\mathrm{j}}^{cv}$ and $\mathrm{H}_{\mathrm{j}}^{dv}$ (Eq. 2.9).

The present value of revenue and costs for conifer and deciduous harvest volume calculation are similar to the one that applied in linear programming model.

Equation 2.14 represents sequential flow policy constraints that resume coniferous and deciduous harvest volume in each time period within 5 % of the previous period's volume. Without sequential flow constraints the model would harvest most of the harvest volume in the first period to maximize net present value. The future forest condition, while dependent on many factors, is strongly influenced by harvest patterns, intensity and schedules. It spatially and temporally presents how the integration of environmental, economic, and social values will be achieved. Determining the planned harvest sequence is imperative to achieving the predicted future forest; thus, 5% sequential flow constraints are included in the model (ESRD, 2012).

The adjacency constraint limits the opening size, which is often required for forest planning. Moreover, adjacency constraint will help to predict natural regeneration such as seeding-in from adjacent stands through wind, birds, or animals (ESRD, 2012). The adjacency constraint (Eq. 2.15) ensures that if stand i is also harvested in period j, none of the stands adjacent to stand i is harvested in period j. The adjacency matrix is output by ArcGIS as two columns. Each line in the output file represents a pair of stands that share at least one point. There are two records in the adjacency file for each matching pair units are indicating that the first stand i is adjacent to the second, and the other represents that the second stand is adjacent to the first. Duplication of adjacent stands was removed. If stands are adjacent, it is assigned as 1, otherwise 0. The mixed integer programming model needs stands if they are adjacent; thus, stands assigned 0 are removed from the table.

The model is designed to choose levels of the decision variables that maximize net present value while satisfying all of the constraints. The system used in this section consists of a mixed integer programming model focused on a timber harvesting problem in four time periods. The objective function (Eq. 2.12) is designed to maximize net present value of the forest plan.

The form of the objective function is as follows:

$$\max Z = \sum_{i=1}^{N}\sum_{j=1}^{T} PF_{ij}X_{ij} \tag{2.12}$$

where Z is the net present value for i = *1,...,N*. The coefficient of net present value is a calculated parameter in the model that is represented in Equation 2.7.

The resource constraint is calculated as follows:

$$\sum_{j=1}^{T} X_{ij} \leq 1 \qquad \forall \quad i = 1, ..., N \tag{2.13}$$

The policy constraints are calculated as follows:

$$\begin{aligned}
&(H_j^{cv}) - 0.95(H_{j-1}^{cv}) \geq 0 \qquad \forall \quad j = 2, ..., T\\
&(H_j^{cv}) - 1.05(H_{j-1}^{cv}) \leq 0\\
&(H_j^{dv}) - 0.95(H_{j-1}^{dv}) \geq 0 \qquad \forall \quad j = 2, ..., T\\
&(H_j^{dv}) - 1.05(H_{j-1}^{dv}) \leq 0
\end{aligned} \tag{2.14}$$

The adjacency constraint is calculated as follows:

$$\begin{aligned}
\sum_{k}^{N} adj_{ik}X_{kj} \leq 1, \qquad &\forall \quad j = 1, ..., T\\
&\forall \quad i = 1, ..., N
\end{aligned} \tag{2.15}$$

2.4 Goal Programming Model Formulation

Another extension to linear programming is goal programming. The goal programming model states the optimization problem as minimizing the aggregate sum of the individual positive and negative weighted deviations from specific goals (Field, 1973). The input data and code of the model are in the appendix C and University of Alberta Education and Research Archive (`http://hdl.handle.net/10402/era.38715`) The goal programming model in this study is based on a mixed integer programming model, focuses on the problem of determining an optimal allocation of scarce resources to meet a given set of multiple objectives, and seeks a plan that comes as close as possible to attaining specified goals. Goal programming models use different weights to attain target values by minimizing the deviations. The significant difference between goal programming and mixed integer programming is that a solution to MIP requires the fulfillment of constraints; however, these constraints are set as goals in goal programming models (Bettinger et al., 2010). Many constraints in a standard linear programming or mixed integer programming model represent management goals of the decision maker. This is done because mathematical programming requires exactly one objective function. Another way of formulating the decision problem would be to use the objective function to represent deviations from specified targets for the management goals. This formulation is called goal programming.

Non-preemptive goal programming provides a way of striving toward multiple objectives simultaneously. The basic approach is to establish a specific target value for each of the objectives and then to seek a solution that gives a good balance towards achieving each of these goals. Penalty weights are assigned to the objectives to measure the relative seriousness of missing their numeric goals (Sherali and Soyster, 1983).

The main advantage of goal programming is its computational efficiency, provided that the target values are known. A goal programming model would be inefficient if target values were set wrong, making a feasible region difficult to approach. In other words, the main disadvantage of goal programming is difficulty of setting penalty weights, objectively. Another advantage of multiple goal programming is that it does not require a very sophisticated solution procedures. Especially the linear goal programming problems can be solved by easily available linear programming routines (Field et al., 1980).

The binary decision variables, X_{ij}, represent whether stand i is harvested in period j or not. Stand i is harvested in period j if the decision variable X_{ij}=*1*. If X_{ij}=*0*, stand i is not harvested in period *j*. The total coniferous and deciduous harvest volumes in each period of the planning horizon in the model are represented as accounting variables, $\mathrm{H}_{\mathrm{j}}^{cv}$ and $\mathrm{H}_{\mathrm{j}}^{dv}$ (Eq. 2.9). The resource constraint formulation is represented in Equation 2.13 and accounting constraints are calculated as in Equation 2.9. Net present value is included in the model as an accounting variable (Eq. 2.17). The mixed integer goal programming model has goal variables that are the deviations of each goal and have to be non-negative in the model.

The present value of revenue and costs for conifer and deciduous harvest volume calculation are similar to the one that applied in linear programming model.

The policy constraints represent secondary goals of the decision makers and the primary goals are in the objective function. In the mixed integer goal programming model, instead of policy constraints, they are converted to the goal constraints. However, the adjacency constraint is placed in the model as a constraint instead of a target which is a same formulation of the one used in mixed integer programming model.

Equation 2.18 represents an example of goal constraints that shows how negative and positive deviations of goals are calculated in the model.

The objective function of the mixed integer goal programming (Eq. 2.16) is minimized the weighted deviations among the harvest levels and net present value. Each target can be weighted in the objective function, which represents the relative importance to this decision makers.

The objective function is calculated in equation 2.16 and each parameters and variables are described in Table 2.1 and each variable is in Table 2.2.

$$\begin{aligned} \min Z = \Big(\sum_{j=1}^{T} uhcw_j PHC_j + ohcw_j NHC_j + uhdw_j PHD_j + ohdw_j NHD_j\Big) \\ + unwPN + onwNN \end{aligned} \tag{2.16}$$

Table 2.1: The description of parameters in objective function

Parameters	**Description**
uhcw$_j$	Under-achievement penalty weight for conifer harv. vol. in period j
ohcw$_j$	Over-achievement penalty weight for conifer harv. vol. in period j
uhdw$_j$	Under-achievement penalty weight for deciduous harv. vol. in period j
ohdw$_j$	Over-achievement penalty weight for deciduous harv. vol. in period j
unw	Under-achievement penalty weight for net present value
onw	Over-achievement penalty weight for net present value

Table 2.2: The description of variables in objective function

Variables	**Description**
NHC$_j$	Negative deviation of conifer harvest volume in period j
PHC$_j$	Positive deviation of conifer harvest volume in period j
NHD$_j$	Negative deviation of deciduous harvest volume in period j
PHD$_j$	Positive deviation of deciduous harvest volume in period j
NN	Negative deviation of net present value
PN	Positive deviation of net present value

The net present value as an accounting constraint is calculated as follows:

$$\sum_{j=1}^{T}\sum_{i=1}^{N} PF_{ij}X_{ij} - npv = 0 \tag{2.17}$$

The mixed integer goal programming model has goal constraints that contain goal variables. The goal variables measure the deviations between management objective levels. The goal variables fill the gap between the goal and what is actually achieved (Buongiorno and Gilless, 2003). Equation 2.18 represents the goal constraint of the net present value which is an example of how negative and positive deviations of goals are calculated in the model.

$$npv - PN + NN = npvgoal \tag{2.18}$$

The harvest volume goal constraint:

$$H_j^{cv} - PHC_j + NHC_j = cgoal_j \qquad \forall \quad j = 1, ..., T \tag{2.19}$$

The conifer volume goal constraint:

$$H_j^{dv} - PHD_j + NHD_j = dgoal_j \qquad \forall \quad j = 1, ..., T \tag{2.20}$$

PN and *NN* represent positive and negative deviation of net present value, respectively. *npv* is the net present value calculation of harvest schedule and *npvgoal* is the goal of net present value that the model tries to reach. *cgoal* (Eq. 2.19) and *dgoal* (Eq. 2.20) are the target values for coniferous and deciduous harvest volumes for each time period.

$$X_{ij}, H_j^{cv}, H_j^{cv}, PN, NN, PHC_j, NHC_j, PHD_j, NHD_j \geq 0 \qquad \begin{array}{l} \forall \quad i = 1, ..., N \\ \forall \quad j = 1, ..., T \end{array} \tag{2.21}$$

The non-negativity constraints represented in equation 2.21 ensure that all of the variables in the model take non-negative values.

2.5 Model Application

Mixed integer programming and goal programming models were applied to a small area of forest, Canada which occupies roughly 1,725 ha, located in Alberta, Canada. The study area is covered mostly by deciduous species. The dominant hardwood species include trembling aspen (*Populus tremuloides*), balsam poplar (*Populus balsamifera*) and white birch (*Betula papyrifera*) while conifer species include black spruce (*Picea mariana*), jack pine (*Pinus banksiana*), white spruce (*Picea glauca*) and tamarack (*Larix laricina*). Table 2.3 shows the distribution of the cover type of the species in hectares. The mixed integer programming model developed in this study was intended to represent a timber harvesting schedule to provide wood flow and maximum net present value. The goal programming model was applied to minimize the weighted deviations of the multiple objectives.

Table 2.3: Broad Cover Type Distribution

Broad Cover Type	**Area (ha)**
Conifer	417.90
Conifer - Deciduous	11.96
Deciduous	891.50
Deciduous - Conifer	29.14
Non-Forest Area	374.77

Two processing facilities, a sawmill and a pulp mill, are represented. The harvest level was determined for a selected harvest schedule. Linear programming of the timber harvest schedule model was used to determine the maximum net present value. Harvested areas were assumed to be replanted immediately after harvesting occurs. Different factors might limit the supply of timber in distinct timber supply areas. For instance, the productivity of the land may be the determining factor in one area, while the age of existing stands may be important in another. Thus, specific management activities, such as harvesting, maintenance, etc., may influence the short or long-term timber supply more in some areas than in others. The age distribution of the study area is shown in Table 2.4.

In this study, yield curve was not used to estimate forest growth; thus, the tables (see appendix A) show stand volume (m^3/ha) was considered constant over all time periods. The minimum duration, or length, of harvest schedule plan is shorter than in a classic forest management plan. The length of 20 years was adopted to create a harvest schedule for four five-year periods.

2.6 Results and Discussion

2.6.1 Linear Programming Model

The objective function, maximizing net present value, is calculated by the differences between discounted cost and revenue. Cost value included haul cost

Table 2.4: Age Distribution

Age (years)	**Area (ha)**
50	4.48
60	15.89
65	61.48
70	453.19
80	14.87
85	255.05
90	247.26
100	3.91
105	107.19
110	173.30
120	12.416
130	1.44

(\$0.0273/m^3/km) and harvesting cost (\$3000/ha). Revenue values came from the determined timber harvest schedule with the timber mill-gate price for both conifer and deciduous species which is \$100/m^3. The maximum net present value obtained from linear programming was \$11.098 million. When the model ran with even-flow constraints, the conifer harvest volume was 21,276 m^3 for each time period, and 31,519 m^3 for deciduous harvest volume for each time period. The even-flow constraints were converted to the sequential flow constraints, resulting in net present value of \$11.3 million. Conifer harvest volume distributions for the successive time periods were 22,940 m^3, 21,793 m^3, 20,703 m^3, and 19,668 m^3. The deciduous harvest volume in each time periods were 33,983 m^3, 32,284 m^3, 30,670 m^3, and 29,137 m^3. The harvest schedule plan resulting from linear programming with even-flow constraints is representing in Figure 2.1.

2.6.2 Mixed Integer Programming Model

Maximum net present value was \$8.6 million with adjacency constraints. Using mixed integer programming, the conifer harvest volume of each period was 17,399 m^3, 16,541 m^3, 15,714 m^3, and 14,937 m^3, and deciduous harvest volume was 23,448 m^3, 24,575 m^3, 24,604 m^3, and 25,823 m^3, respectively. The sequential flow constraints in the model were satisfied with 5% wood flow between each time

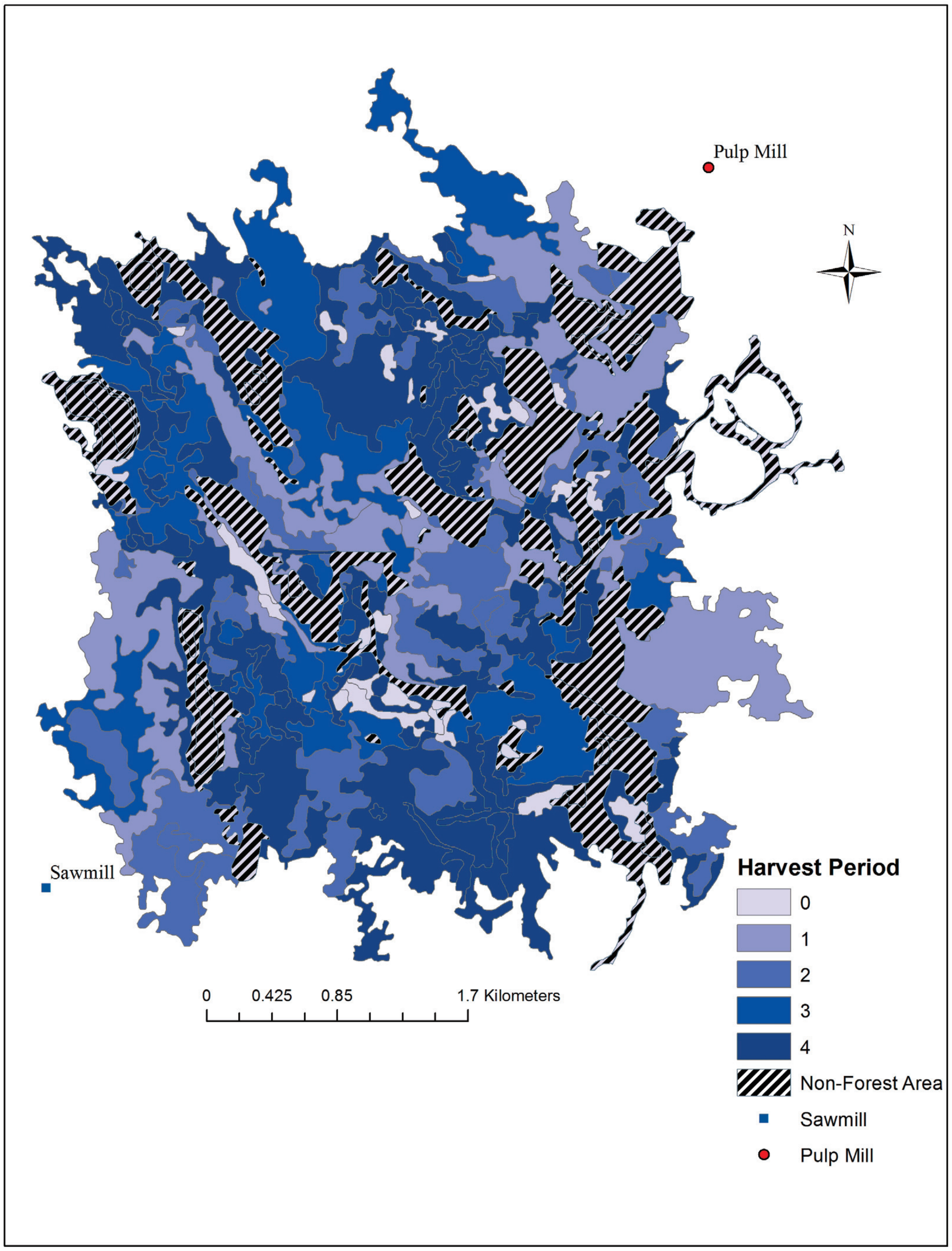


Figure 2.1: Harvest schedule map produced by LP model

period. The model also was run without adjacency constraint being considered. The net present value in that situation was \$ 11.3 million. The sequential wood flow for coniferous harvest volumes in successive time periods were 22,952 m^3, 21,808 m^3, 20,720 m^3, and 19,689 m^3 and 33,991 m^3, 32,297 m^3, 30,685 m^3, and 29,153 m^3 for deciduous harvest volumes. The results for the mixed integer programming model found an optimal feasible solution by satisfying sequential flow, resource, accounting and adjacency constraints. Figure 2.2 shows the harvest schedule map resulting from mixed integer programming model.

2.6.3 Goal Programming Model

The objective function in the goal programming model is to minimize the sum of weighted deviations from goals. The goal programming model includes three objectives for each time period: maximum net present value, target volumes of softwood harvest and target volumes of hardwood harvest. The net present value goal was set to \$20 x 10^6 to challenge the goal programming model to reach as high as profit. The harvest volume goals for both conifer and deciduous species were adjusted for the model which are also obtained from the mixed integer programming model. The target value for the coniferous harvest was set 15,000 m^3 harvest volume was assigned. The deciduous harvest volume goal was set at 25,000 m^3 for each time period. Different priorities affect goal achievements such as giving top priority to net present value and harvest volumes. The goal programming model penalized those goals when weights were set (Table 2.5). The weights of objectives are determined according to the idea of relative penalty weight (Gass, 1987). The under-achievement penalty weight of net present value was penalized at 1 penalty unit (pu). The over-achievement and under-achievement penalty weights of conifer harvest volume in period one through four were assigned as 1333 pu which is relative to the net present value target (20x10^6 : 1.5x10^4). The same method was applied to set deciduous harvest volume penalty weights (20x10^6 : 1.5x10^4). The model could find estimated results for each of the goals separately if other goal weights are assigned 0. However, even though the goal programming model produced a feasible solution, solution procedure took a long time and was not improved by solving with GUSEK which ran out of virtual memory to complete the optimization problem. Another optimization problem solver, Gurobi Optimization (2014), was used to solve the mixed integer goal programming model. The best objective function of the goal

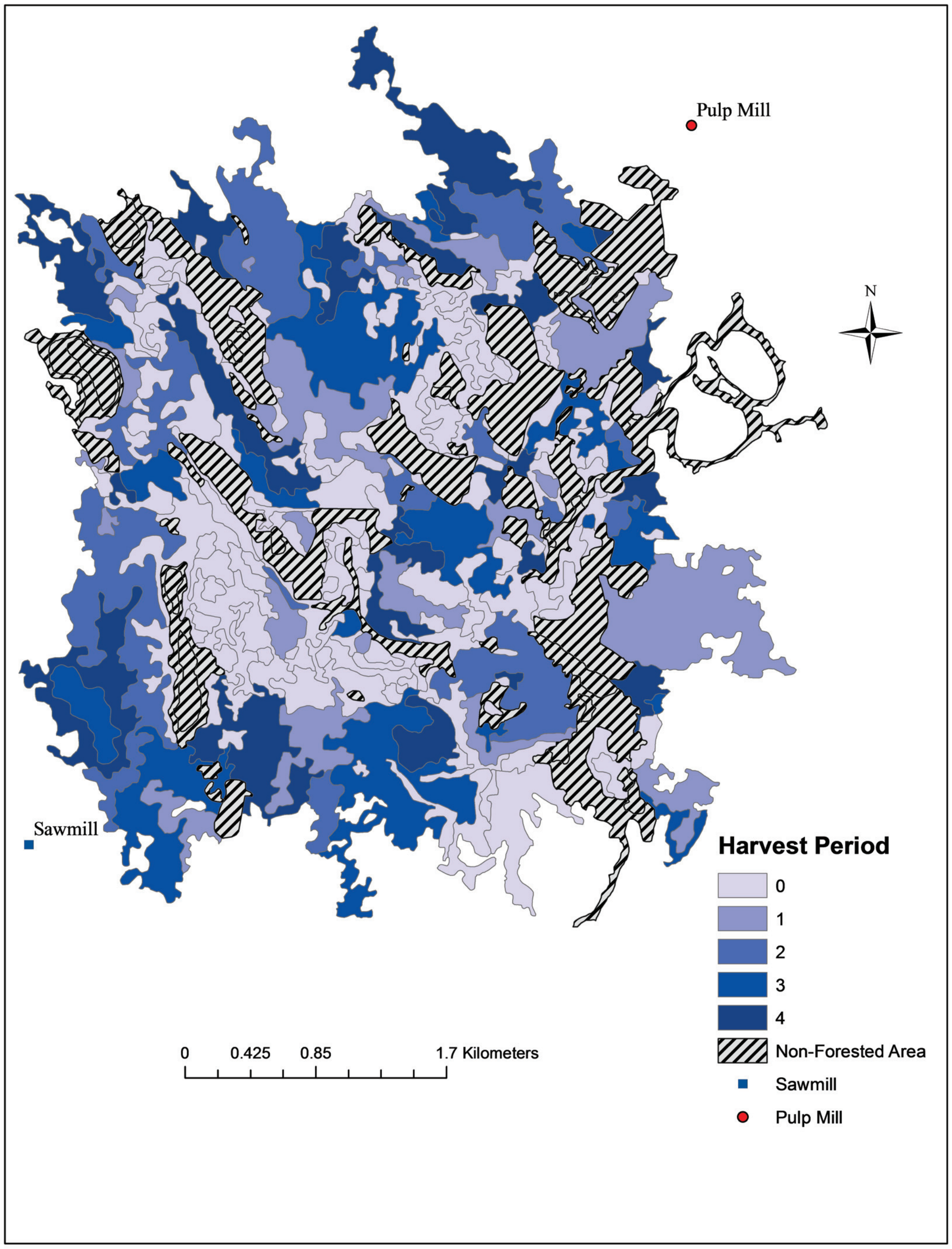


Figure 2.2: Harvest schedule map produced by MIP model

programming model was 1.138 x 10^7, which has 8 continuous, 924 integer variables. The time limit was set to two hours to find a feasible solution, and the model was satisfied with a 0.54% gap. It does not guarantee the optimal solution but does guarantee that the result within 0.54 % of the optimal one. The resulting conifer harvest volumes for each time period were 17,640 m^3, 14,486 m^3, 15,000 m^3, and 15,045 m^3 while the deciduous harvest volumes were 24,788 m^3, 25,000 m^3, 25,000 m^3, and 24,911 m^3, respectively.

Both the mixed integer model and the mixed integer goal model satisfied the adjacency constraints; however, solution time for goal programming as longer than for mixed integer programming. The net present value was \$8.618 x 10^6 which is close to the result of the mixed integer goal programming model. Table 2.6 represents the summary results of the goal programming model. The harvest schedule of the goal programming model is represented in the Figure 2.3. The goal programming model resulted in up an acceptable solution.

The results of three different models with different constraints are compared and represented in Table 2.7. In developing the linear programming model, the adjacency constraints were omitted. The extensions to the other linear programming approaches examined here incorporate the adjacency constraint. When the linear programming model incorporates sequential flow instead of even flow and decision variables are not binary, the net present value and harvest volume for coniferous and deciduous trees in each time period are close to the results derived from the mixed integer programming model without adjacency constraint. Adjacency constraint plays a big role in the model, which impacts net present value and harvest volume in each time period. As shown in Table 2.7, harvest volume of the first period is higher than in other periods because the model maximizes net present value.

Table 2.5: Penalty weights for objective function of the goal programming model

	Goals	**Under-achievement**	**Over-achievement**
NPV	$20x10^6$	1	0
Con. Harv. Vol	$1.5x10^4$	1333	1333
Dec. Harv. Vol	$2.5x10^4$	800	800

Table 2.6: Summary table for goal programming model when all goals are penalized

Penalty	$1.14 \text{ x } 10^7$
Net Present Value ($)	$8.62\text{x } 10^6$
Harvest Volume of Coniferous (m^3)	
Period 1	$1.76 \text{ x } 10^4$
Period 2	$1.45 \text{ x } 10^4$
Period 3	$1.50 \text{ x } 10^4$
Period 4	$1.50 \text{ x } 10^4$
Total Volume	$6.21 \text{ x } 10^4$
Harvest Volume of Deciduous (m^3)	
Period 1	$2.48 \text{ x } 10^4$
Period 2	$2.50 \text{ x } 10^4$
Period 3	$2.50 \text{ x } 10^4$
Period 4	$2.49 \text{ x } 10^4$
Total Volume	$9.97 \text{ x } 10^4$

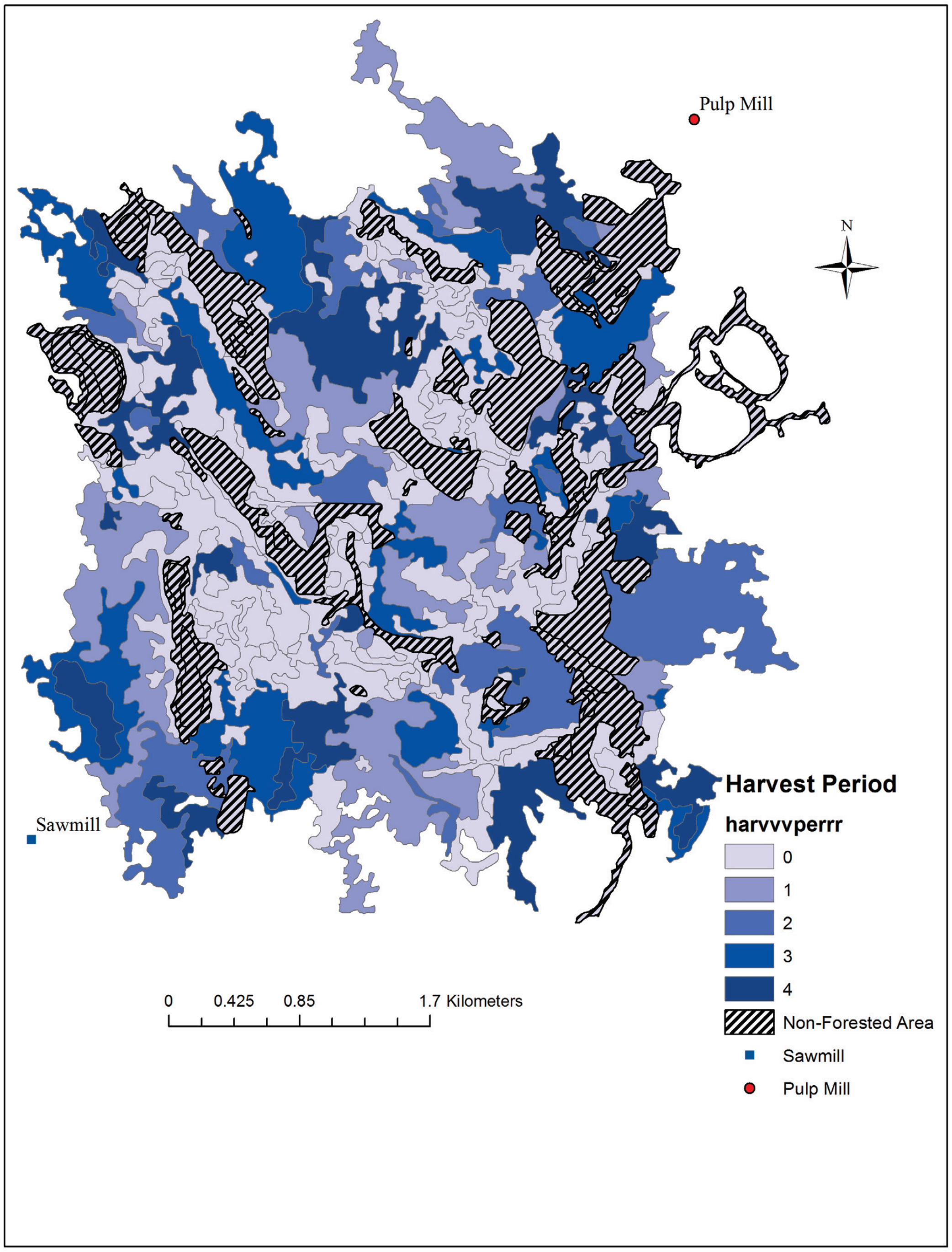


Figure 2.3: Harvest schedule map produced by goal programming model

Table 2.7: Summary results of three models with different constraints

	Linear Programming		**Mixed Integer Programming**		**Goal Programming**
	Even-Flow Constraints	Sequential-Flow Constraints	Adjacency Constraint	No Adjacency Constraint	Adjacency Constraint
Net Present Value ($)	11.1x10^6	11.3x10^6	8.6x10^6	11.3x10^6	8.6x10^6
Con. Harv. Vol. 1 (m^3)	21,276	22,940	17,399	22,952	17,640
Con. Harv. Vol. 2 (m^3)	21,276	21,793	16,541	21,808	14,488
Con. Harv. Vol. 3 (m^3)	21,276	20,703	15,714	20,720	15,000
Con. Harv. Vol. 4 (m^3)	21,276	19,668	14,937	19,689	15,045
Dec. Harv. Vol. 1 (m^3)	31,519	33,983	23,448	33,991	24,788
Dec. Harv. Vol. 2 (m^3)	31,519	32,284	24,575	32,297	25,000
Dec. Harv. Vol. 3 (m^3)	31,519	30,670	24,604	30,685	25,000
Dec. Harv. Vol. 4 (m^3)	31,519	29,137	25,823	29,153	24,911

2.7 Conclusion

The optimization modeling system in this chapter is composed of a linear programming, mixed integer programming and mixed integer goal programming timber supply model, used to determine optimal harvest schedules. Both extensions of linear programming were used in this chapter to provide information and guidance for the genetic algorithm model that is developed and applied in the next chapter. The linear programming models are easy to apply, providing immediate advantages for short and long-term planning periods compared to the empirical methods traditionally used in harvest planning.

In the next chapter, the mixed integer goal programming model is recast as a genetic algorithm model to allocate forest zones (harvesting and reserve zone) and includes some other constraints and complex targets that are difficult to formulate in a mixed integer programming framework.

Bibliography

Bettinger, P., Boston, K., Siry, J., Grebner, D. L., 2010. Forest Management and Planning. Academic Press, San Diego, CA.

Buongiorno, J., Gilless, J. K., 2003. Decision Methods for Forest Resource Management. Academic Press, New York.

Clutter, J. L., 1968. Max-million: a computerized forest management planning system. The University of Georgia. School of Forest Resources. Biometrics-Operations Research Section.

Dykstra, D., 1984. Mathematical programming for natural resource management. McGraw-Hill Series in Forest Resources. McGraw-Hill, New York. 318p.

ESRD, 2012. Alberta timber harvest planning and operating ground rules framework. Alberta Environment and Sustainable Resource Development. Available at: http://esrd.alberta.ca/lands-forests/forest-management /documents/TimberHarvest-OperatingGroundRules-Jun2012.pdf. [Accessed 2014-04-21].

Field, D. B., 1973. Goal programming for forest management. Forest Sci. 19 (2), 125–135.

Field, R. C., Dress, P. E., Fortson, J. C., 1980. Complementary linear and goal programming procedures for timber harvest scheduling. Forest Science 26 (1), 121–133.

Gass, S. I., 1987. The setting of weights in linear goal-programming problems. Computers & operations research 14 (3), 227–229.

Greer, K., Meneghin, B., 1991. Spectrum: An analytical tool for building natural resource management models. US Department of Agriculture Forest Service General Technical Report, 174–178.

Gurobi Optimization, I., 2014. Gurobi optimizer reference manual. Available at: http://www.gurobi.com.

GUSEK, 2013. GLPK Under Scite Extended Kit version 0.2.14. Luiz Bettoni. Boston, USA.

Johnson, K., Jones, D., 1979. A users guide to multiple use sustained yield resource scheduling calculation (MUSYC). Washington, DC: USDA. Forest Service, Timber Management Staff. 242p.

Johnson, K. N., Stuart, T. W., Crim, S. A., 1986. Forplan version 2: An overview. Washington: USDA Forest Service.

Navon, D. I., 1971. Timber RAM A long-range planning method for commercial timber lands under multiple-use management. U.S. Pacific Southwest Forest and Range Experiment Station. USDA Forest Service research paper PSW-70 Berkeley, Calif.

Remsoft Inc, 2002. Spatial Woodstock users guide. Frederiction, Canada.

Sherali, H., Soyster, A., 1983. Preemptive and nonpreemptive multi-objective programming: Relationship and counterexamples. Journal of Optimization Theory and Applications 39 (2), 173–186.

Steuer, R., Schuler, A., 1978. An interactive multiple-objective linear programming approach to a problem in forest management. Operations Research 26 (2), 254–269.

Chapter 3

Genetic Algorithm Model Formulation and Solution

3.1 Introduction

This chapter describes the genetic algorithm model developed for this thesis. The modeling system section will address the methods of problem formulation for a forest planning model. The forest planning problem investigated here attempts to find a near optimal solution using the genetic algorithm technique. In this chapter, the formulation of the genetic algorithm is an extension of the decision problem presented in chapter 2. Unlike the goal programming and mixed integer programming models, the genetic algorithm model is able to count adjacency violations to allow a few stands to be harvested in the same period and to aggregate reserve stands for the purpose of a creating a reserve area comprised of stands close to each other. The solution to this problem will be provided by the model that produces the harvest schedule with the lowest penalty values across all generations. The forest management problem in this study considers twelve goals:

1) Maximize present value ($),

2−5) Reach target softwood harvest volumes in each of periods 1-4 (m^3),

6−9) Reach target hardwood harvest volumes in each of periods 1-4 (m^3),

10) Minimize number of adjacency restriction violations,

11) Minimize the minimum spanning tree distance between reserve stands (m), and

12) Reach target value for quality weighted reserve area.

Essentially, the decision maker desires to obtain maximum net present value while reaching stable target value for conifer and deciduous harvest volume in period 1 through 4, with few adjacency violation, and a created a high quality reserve area made up of stands close to each other.

3.2 Modeling System

A genetic algorithm provides a heuristic optimization method based on the mechanisms of natural selection and evolution. The genetic algorithm operates with a population of possible solutions. Each member in the population represents a harvest schedule. The initial population of solutions is generated randomly and subsequently subjected to simulated genetic evolution over a number of generations. The objective function tries to satisfy multiple conditions and calculates a penalty value for each member in all generations.

Three genetic operators are used in genetic algorithms to generate diversity. Selection is an operator that selects parents to breed to produce the next generation. The probability of a randomly selected individual being selected as a breeder is proportional to its fitness value. The fitness value is measured in relation to the penalty function value in the optimization problem. Individuals with high fitness are more likely to be selected as breeders and the idea is that two parents with high fitness will likely produce a fit child (Shiffman et al., 2012). Following selection, the crossover operator combines two breeders to produce new members. Crossover simulates recombination and allows the children so bred to have a mix of each parent's genes. Mutation is a random alteration of genes in children resulting from a crossover. In this study, all individuals represent a harvesting plan and each gene is a harvesting period (0-4). The mutation probability is usually very small; thus, only a tiny portion of genes mutate. Mutation helps in escaping regions of local optima (Fotakis et al., 2012). The child with the lowest penalty value from each breeding becomes a member of the next generation. This process continues for a number of population.

The model used in this study is an optimizing forest estate model used to choose optimal timber harvesting plans and reserve areas as solved by a genetic algorithm. The model is programmed in MATLAB (2010) shown in Appendix D and also available online at the University of Alberta Education and Research Archive at `http://hdl.handle.net/10402/era.38715`. The structure of the goal programming model in chapter 2 influenced the design of the genetic algorithm model built here. As objective function in the goal programming model, the penalty function minimized the sum of the weighted deviations from specific goals.

Targets were based on the results of the solution to the mixed integer program in chapter 2. Target values for the goal in penalty function are shown in Table 3.1. The maximum net present value can be reached by penalizing the under-achievement weight. Deviations from target harvest volumes for conifer and deciduous for each time period are included in the penalty function. Instead of restricting adjacent stands to zero, violations of adjacency are counted with the idea that some trade off involving a small number of violations could be acceptable. In the genetic algorithm model, the adjacency constraint is converted to a goal, which minimizes the total amount of neighboring stands that are harvested in the same time period. Each stand in the harvesting schedule plan is assigned a value between 0 and 4. The numbers 1 through 4 represent harvesting periods and 0 means reserve stands (stands not harvested) in the forest plan. The reserve quality of these reserve stands is related to forest age. The minimum spanning tree is the shortest path that connects all stands selected as reserve areas. Using the minimum spanning tree algorithm is conservation strategy that I chose to group reserve stands in this study. The objective function in the genetic algorithm calculates the penalty values of members which is the sum of the weighted deviations of goals. In the model developed here, a forest plan is represented by a vector, h, of length N where N is the number of stands in the forest. Each element of the vector, h_n, represents the harvest period, for stand i; where the value of zero represents no harvesting activity. The optimal harvest schedule conceivably could be found through complete enumeration. However, this would be impractical for any realistically sized problem. Even with the small problem examined here with 5 possible harvest periods and 287 stands, there would be 5^{287} (4.02x10^{200}) possible solutions to examine.

Table 3.1: Table of target values for goals in penalty function

Objectives	**Target Values**	**Units**
Net Present Value	11.3x10^6	\$
Conifer Harvest Volume Period 1	1.5x10^4	m^3
Conifer Harvest Volume Period 2	1.5x10^4	m^3
Conifer Harvest Volume Period 3	1.5x10^4	m^3
Conifer Harvest Volume Period 4	1.5x10^4	m^3
Deciduous Harvest Volume Period 1	2.5x10^4	m^3
Deciduous Harvest Volume Period 2	2.5x10^4	m^3
Deciduous Harvest Volume Period 3	2.5x10^4	m^3
Deciduous Harvest Volume Period 4	2.5x10^4	m^3
Adjacency Violation	0	count
Minimum Spanning Tree	0	m
Reserve Value	200	quality-weighted ha

At the beginning of the model, the initial population of harvest schedule is created randomly. The initial population includes 100 forest plans. One hundred plans are selected for breeding and produce two candidate plans. Penalty values are calculated for each plan based on weighted deviations from goals and these are used to calculate fitness. Low penalties correspond to a high fitness value and vice-versa. Selection of the potential breeding parents is with the probability of the selection proportional to the fitness of the parents. The fittest of the two child plans is selected for inclusion in the next generation. For each generation, a new breeding population is determined and the current population is replaced by the population of children. The fitness value for each forest plan plays a significant role in the algorithm because it changes the structure of the population and directs the model to an optimal solution. Using the fitness value is a probabilistic method to choose best breeders. A fitness score is assigned to each solution representing the abilities of an individual to compete for breeding opportunities.

Net present value is the sum of the discounted cash flow. In this study, harvest revenue is an income calculated as a dollar value from stands that are assigned to harvest (one of the four harvest periods). Each stand in the reserve zone has a reserve quality score that is a function of the stand age. The purpose of the reserve quality is to select forest areas that best meet conservation goals. In the model presented here, I assume that older forest has a higher conservation value. The reserve value, the quality weighted area, is the sum of the product of the reserve stand's index quality and the stand areas. Some spatial considerations are required for multiple objective forest planning. Adjacency constraint is spatial requirements is adjacency constraints that restrict the selected harvesting areas so that neighbor stands cannot be harvested in the same period. In this algorithm, adjacent stands that are harvested in the same period are counted. The adjacency violation term is a penalty function that tends to minimize the count of adjacency violations.

3.2.1 Genetic Algorithm Model Formulation

The harvest scheduling problem in this study attempts to find the best spatial and temporal arrangements of treatments to achieve multiple objectives. The penalty is proportional to the extent of the violations. The penalty function makes the model favor solutions with fewer and less severe deviations from goals (Mullen, 2000). The penalty function of the genetic algorithm is adopted from the goal programming objective function in chapter 2.

The most important and difficult part of goal programming is the determination of appropriate goal weights (Gass, 1987). The penalty function, *P*, is used to calculate weighted deviations from goals for a given harvest schedule, *P(H)*. A high penalty value represents large deviations from the goals while small penalty values are smaller deviations from the goals. The objective of the genetic algorithm is to find the harvest schedule that has the lowest penalty- the smallest sum of weighted deviations from the target. The goal constraints used in the integer goal program from chapter 2 were extended and included in the genetic algorithm as the penalty function. The penalty function minimizes the deviations from the target values that have been specified for objective variables, usually at the planning area level (Kangas et al., 2008).

The model formulation of the harvest scheduling problem using a genetic algorithm model is defined as follows:
The penalty function, *P(H)*, is calculated for each candidate harvest schedule H as follows:

$$
\begin{aligned}
P(H) =& \Big(\sum_{j=1}^{T} uhcw_j NHC_j + ohcw_j PHC_j + uhdw_j NHD_j + ohdw_j PHD_j\Big) + \\
& unwNN + onwPN + uawNA + oawPA + umwNM + omwPM + \\
& utwNT + otwPT
\end{aligned}
\tag{3.1}
$$

Each weight in Equation 3.1 is described in Table 3.2 and each variable is in Table 3.3.

Table 3.2: The description of parameters in objective function

Parameters	**Description**
uhcw$_j$	Under-achievement penalty weight for conifer harv. vol. in period j
ohcw$_j$	Over-achievement penalty weight for conifer harv. vol. in period j
uhdw$_j$	Under-achievement penalty weight for deciduous harv. vol. in period j
ohdw$_j$	Over-achievement penalty weight for deciduous harv. vol. in period j
unw	Under-achievement penalty weight for net present value
onw	Over-achievement penalty weight for net present value
uaw	Under-achievement penalty weight for adjacency violation
oaw	Over-achievement penalty weight for adjacency violation
umw	Under-achievement penalty weight for minimum spanning tree
omw	Over-achievement penalty weight for minimum spanning tree
utw	Under-achievement penalty weight for reserve value
otw	Over-achievement penalty weight for reserve value

Examples of the negative and positive deviations of goals is calculated in equation 3.2 and 3.3 as:

$$PN = \max(0, npv - npvgoal) \tag{3.2}$$

$$NN = \max(0, npvgoal - npv) \tag{3.3}$$

where NN as a negative deviation for net present value is found by net present value of selected harvest schedule (*npv*) minus desired net present value goal (*npv-*

Table 3.3: The description of variables in objective function

Variables	**Description**
NHC_j	Negative deviation of conifer harvest volume in period j
PHC_j	Positive deviation of conifer harvest volume in period j
NHD_j	Negative deviation of deciduous harvest volume in period j
PHD_j	Positive deviation of deciduous harvest volume in period j
NN	Negative deviation of net present value
PN	Positive deviation of net present value
NA	Negative deviation of adjacency violation
PA	Positive deviation of adjacency violation
NM	Negative deviation of minimum spanning tree
PM	Positive deviation of minimum spanning tree
NT	Negative deviation of reserve value
PT	Positive deviation of reserve value

goal). Unlike negative deviations of the goals, a positive deviation is computed by subtracting the calculated value of harvest schedule from the assigned goal.

The goal variables calculated in the model as follows:

The net present value for each harvest schedule in the population is found by using following formula:

$$npv = \sum_{j=1}^{T} \frac{sc((cv\ a\ pc + dv\ apd) - (a\ hc) - (cv\ d_{is}\ k + dv\ d_{ip}\ k))}{(1+I)^{m_j}} \tag{3.4}$$

where sc represents harvest schedule and cv and dv are matrices for stand volume (m^3/ha). The price paid per m^3 of conifer and deciduous are represented as pc and pd (\$/ha). The matrices of the area is a (ha). The cost of harvesting is hc (\$/ha) and k is the haul cost (\$/m^3/km). The distance (km) between stand i and the sawmill matrix is represented in the equation as d_{is} while the distance between stand i and the pulp mill matrix is d_{ip}. The discount rate is represented as I and m_j is the midpoint of the harvested period in years (Eq. 3.4).

$$Hv^c = sc\ cv \tag{3.5}$$

$$Hv^d = sc\ dv \tag{3.6}$$

Harvest volumes for conifer (Hv^c) and deciduous (Hv^d) species are found by multiplying stand volume by harvest schedule, *sc*, where each row represents stand and each column represents harvest periods as shown in equation 3.5 and 3.6.

$$Av = sc.at.sc' \tag{3.7}$$

In the goal programming and mixed integer programming models illustrated in chapter 2 no adjacency violations were permitted. Adjacency violation is calculated in this model by summing the total number of neighbor stands that are harvested in the same time period. In the genetic algorithm model used here, adjacency violations are counted and penalized. Equation 3.7 shows the calculation of adjacency violation in the genetic algorithm model where *at* represents adjacent stands as a matrix. The *at* is an adjacency matrix that is created by the analysis tool in ArcGIS (Maene, 2011). Adjacency matrices are identified if a stand is adjacent to another stand by sharing at least one point in their boundaries. Multiplying the lower triangular part of adjacency matrix (*at*) and harvest schedule (*sc*) and conjugate transpose matrix of harvest schedule creates a table for adjacency violations. The sum of the diagonals of this matrix gives adjacency violation for a selected harvest schedule.

One planning goal is to group selected areas into a reserve zone by calculating a minimum spanning tree. The single large or several small (SLOSS) was a debate as to whether a single large or several small reserves are a better means of conserving biodiversity. Single large reserves, even when if several small reserves can cover the same total area, have been recommended as the most effective means of protecting endangered populations (Diamond, 1975). I chose to use minimum spanning tree algorithm as an expression of the level of aggregation of the reserve stands. The idea is that reserve stands located close to each other would be more effective for conservation purposes than stands scattered randomly throughout a management area. The minimum spanning tree needed to connect reserve stands was calculated for the graph by using Prim's (1957) algorithm, which minimizes the distance of each stand's centroid.

$$
\begin{aligned}
G = min \sum_e E c_e x_e \\
s.t.: \sum_e E x_e = n-1 \qquad \forall \quad n = 1, ..., V
\end{aligned}
\tag{3.8}
$$

The minimum spanning tree function (G) with cost c_e for all of the edges in E finds a spanning tree *G(E, V)* of minimum total cost. x_e is the decision variables, if edges e is element of E, the cost is assigned as 1; otherwise 0 (Eq. 3.8). The minimum spanning tree contains the set of links of minimum total cost (summed distance in meters) that joins all stands into a single connected cluster. The grouped reserve stand is based on the minimum spanning tree, which is the spatial equivalent to the dendrogram of the stands. The reserve stands are meshed by Delanuay triangulation. The Delaunay triangulation described in Smaltschinski et al. (2012) as for a set of points in a plane, is one that maximizes the minimum angle of all the triangles in the triangulations. In other words, Delaunay triangulation finds links between stand centroids and Prim's algorithm selects the links that make up the minimum spanning tree. Prim's algorithm, modelled by Kundur (2005), was used to calculate the minimum spanning tree cost (distance between stands, in this study). Applying Delaunay triangulation to fill the distance matrix reduces the distance calculation time. Without Delaunay triangulation, the minimum spanning tree would consider every possible links between centroids of each stand that takes longer to create minimum spanning tree graph. In the minimum spanning tree function, Delaunay triangulation is created by using a Matlab built-in function from a set of points, which ensures that the circumcircle associated with each triangle contains no other point in its interior. Prim's algorithm uses an adjacency matrix that was generated by Delaunay triangulation. This adjacency matrix is n by n, where n is select reserve stands. The procedure that provides optimal solution for reserve areas is to select the shortest possible link to any other stand without considering the effect this might have on following operations (Dykstra, 1984).

Prim's algorithm is an algorithm in graph theory that finds a minimum spanning tree for connected undirected weighted graph. In this case, weights are Euclidean distances between stand centers. The algorithm starts with the first node and keeps track of which nodes are in trees and which are not. The algorithm iterates until all nodes are in trees. The function gives the cheapest edge from a node that is in the tree to one that is not. Prim's algorithm builds the tree by adding leaves to the current tree one at a time. The edges of the minimum spanning tree are returned in array *mst* (of size n-1 by 2), and the total cost is returned in the variable cost corresponding to distance in meters (Prim, 1957). Figure 3.1 shows a selected reserve area connected each other with Delaunay triangulation, and these connections between stands are linked each other with Prim's algortihm.

$$rv = \sum_{i=0}^{N} sc_j a_i sqi \qquad j = 0 \in T \tag{3.9}$$

Reserve value (rv) represents total reserve value, which is found by multiplying reserve stands that are assigned as 0 in the harvest schedule (sc_0) by its area (a_i) and reserve quality index (sq_i) (Eq. 3.9).

3.2.2 Genetic Algorithm Operators

A genetic algorithm model generates solutions to optimization problems using techniques inspired by natural evolution. The evolution starts from a population of randomly generated individuals. The fitness of every harvest plan in the population is evaluated. Fitness is the value of the penalty function in the optimization problem. Each harvest plan in the population is carrying fitness value. *f(i)* is the fitness of individual i and is used to determine the probability of an individual's being selected. The population in the each generation needs to be evaluated to determine which harvest plans are fit to be selected as parents for the next generation. The fitness function produces a numeric score to describe the fitness of a member of a population (Shiffman et al., 2012). The desirable result for the genetic model in this study is the lowest penalty in the population of each generation. The fitness score is calculated by dividing the differences between the maximum penalty and the penalty of the selected harvest plan in the population to the penalty array that is comprised of the differences between maximum and minimum penalties in the population.

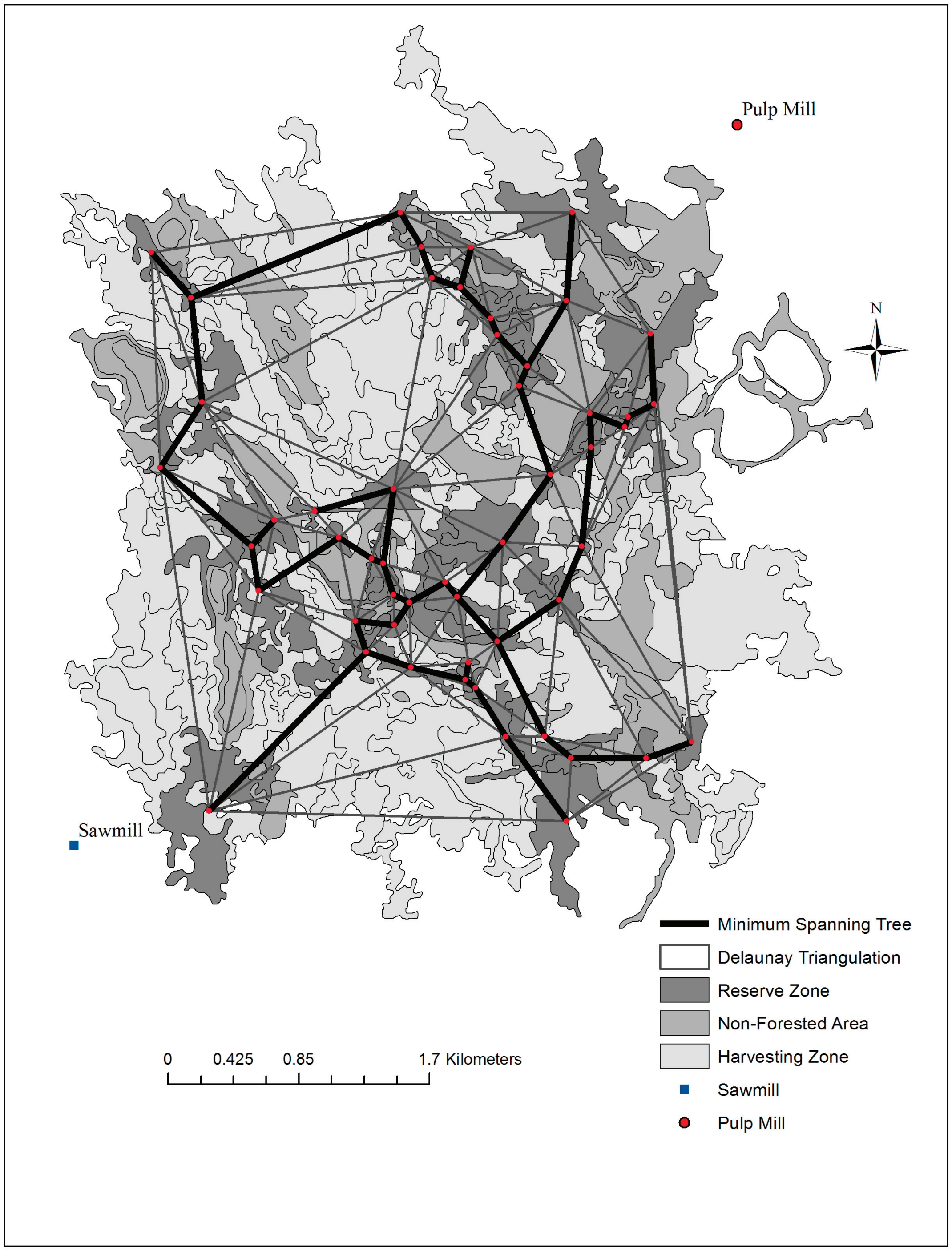


Figure 3.1: Minimum spanning tree function selects a set of Delanuay triangulation that minimize sum of the node lengths

The fitness value is calculated in the function as follows:

$$f(i) = \frac{\max(P) - P_i}{\max(P) - \min(P)} \tag{3.10}$$

The fitness value of i is calculated by fitness function, *f(i)*, where P_i is the penalty value of the individual i, *max(P)* is assigned as a maximum penalty and *min(P)* is the minimum penalty of the generation (Eq. 3.10).

The next step is to produce next generation by applying a combination of genetic algorithm operators: selection, crossover and mutation.

Selection: The genetic algorithm model calculates the penalty value for each harvest plan in the population. Some of harvest plans of the population have the opportunity to be breeders and pass their genetic information to next generation. The population needs to be evaluated to determine which harvest plans are selected as parents for the next generation (Shiffman et al., 2012). The pseudo-code of the selection operator is shown in Figure 3.2. Each harvest plan in the population has a chance to be selected associated with its fitness value. The selection operator chooses two harvest schedules as breeders by comparing the fitness value of an individual harvest plan with a random number between 0 and 1. The fittest member in the population, represented by *"1"*, has the highest chance to be selected as a breeder.

```
while breed <2
  breed=random([sc],1,2) % Randomly selects two potential breeder
  %compare fitness value of breeders and random number
   between 0 and 1
  if fit(breed(1)) >= randomchance[0,1];
     parents(1) = breed1;
  elseif fit(breed(2)) >= randomchance[0,1];
     parents(2) = breed2
   end
end
```

Figure 3.2: Pseudo-code of selection operator

For each member of the next generation, the algorithm selects potential breeding pairs. Two uniformly distributed random numbers between zero and one are drawn and compared to the fitness values for the two potential harvest plans as breeders. If the values for both potential breeders are greater than the corresponding random numbers, the pair breeds, otherwise a new potential breeding pair is selected.

Crossover: Crossover is a genetic operator to use to introduce variation to the next generation by creating a new harvest plan out of the genetic code of the harvest plans that are selected as breeders from the previous population. In other words, crossover shuffles the genes of two breeders to create two offspring showing some of the characteristics better adapted to the environment. In this study, environment is the goals and the penalty weights.

$$\begin{aligned} ch_1 &= p_1(1 : xr), p_2(xr : n) \\ ch_2 &= p_2(1 : xr), p_1(xr : n) \end{aligned} \tag{3.11}$$

Crossover produces two candidates (*ch1* and *ch2*) for a new generation which are offspring produced by a mix of their parent's genes. *xr* is a random integer between 1 and *n*-1, where *n* is the number of stands, to choose which part of the parents genes are selected to produce new members (Eq. 3.11). Crossover occurs once for each individual.

Mutation: The last operator of the genetic algorithm is mutation, which mutates children if they are eligible. The mutation operator is applied to each gene of each children resulting from the crossover operation of a selected harvest plans. Each forested stand (or genes) in the individual harvest schedule has a small probability of mutating. A randomly selected number between 0 and 1 is assigned for each gene (harvest period) in the individual (harvest plan). If this randomly generated number is smaller than the mutation rate, the gene (harvest period) mutates. In the model presented here, a mutation represents a random change in the harvest period of a harvest schedule candidate. Mutation has been used in genetic algorithms to help the algorithm move away from a local optimum in the search for a global optimum. It also adds some diversity to the population which would likely prove useful in the case of adapting to the environment. The mutation operator is explained as a pseudo-code in Figure 3.3. One hundred new harvest plans are added to the new population. For the next generation, breeders are selected from the previous population. The process used by the genetic algorithm to generate a harvest schedule is shown in Figure 3.4 as a pseudo-code.

```
for i=n:number of stands
  if random[0,1] <= mutation rate; %Random number to compare with
    newmutate = random([0,1],nstands) %New pair to mutate with
    child1(i) = newmutate(i)
  if random[0,1] < mutation rate;
    newmutate = random([0,1],nstands)
    child2(i) = newmutate(i)
  end
end
```

Figure 3.3: Pseudo-code of mutation operator

```
  #Initialization
  for i=n:number of member  in the initial population
    generate randomly initial population;
    set as Z;
    calculate hv(i); %harvest volume
    calculate npv(i); %net present value
    calculate adv(i) ;%adjacency violation
    calculate P(i); %penalty value
    calculate f(i) %fitness value
       (max P - P(i)) / (max P - min P);
  end
  initialpop=oldpop
  for i=n:number of generation
   newpop=[]
     #Selection
     #Crossover
     #Mutation for child1 & child2
    hv(i); npv(i); adv(i); P(i);
    if P(child1) < P(child2)
       Z(i) = [Z(i); child1]
    else
       Z(i) = [Z(i); child2]
    end
    minpenalties=[minpenalties, P(min) in Z(i)];
    oldpop=newpop;
  end
```

Figure 3.4: Pseudo-code of genetic algorithm for harvest scheduling

3.3 Model Application

The model was applied to a small forest area of 1,725 ha located in Alberta, Canada (Fig.3.5). Of the total area, 1350 ha (or 231 out of 287 stands) have tree cover and are eligible for harvest. Figure 3.6 shows the dominant species in the area is deciduous. The dominant hardwood species include trembling aspen (*Populus tremuloides*), balsam poplar (*Populus balsamifera*) and white birch (*Betula papyrifera*) while conifer species include black spruce (*Picea mariana*), jack pine (*Pinus banksiana*), white spruce (*Picea glauca*) and tamarack (*Larix laricina*). Table 3.4 shows the distribution of the cover type of the species in hectares.

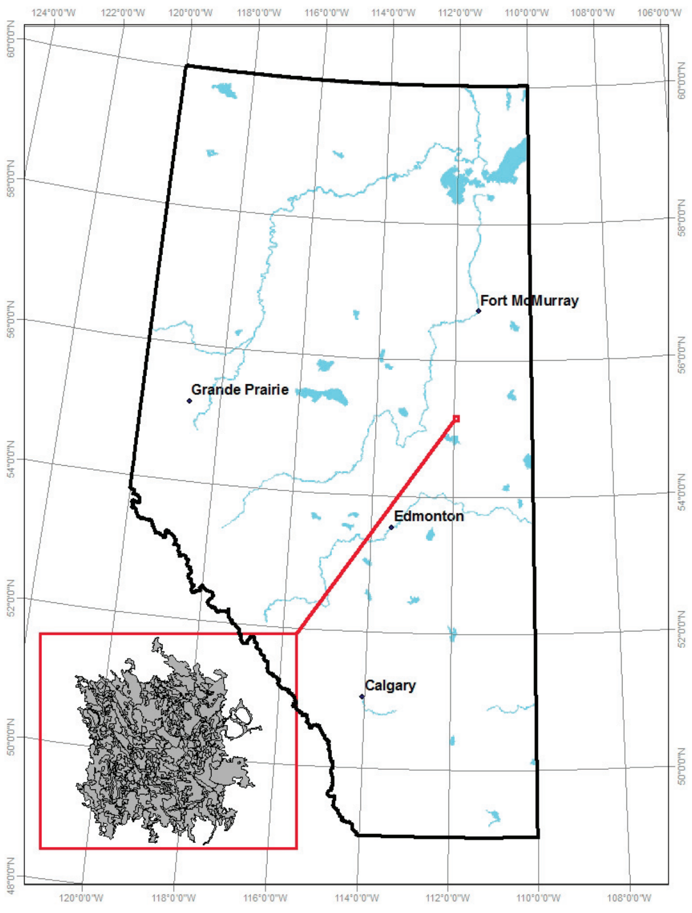


Figure 3.5: Location of study area within Alberta, Canada

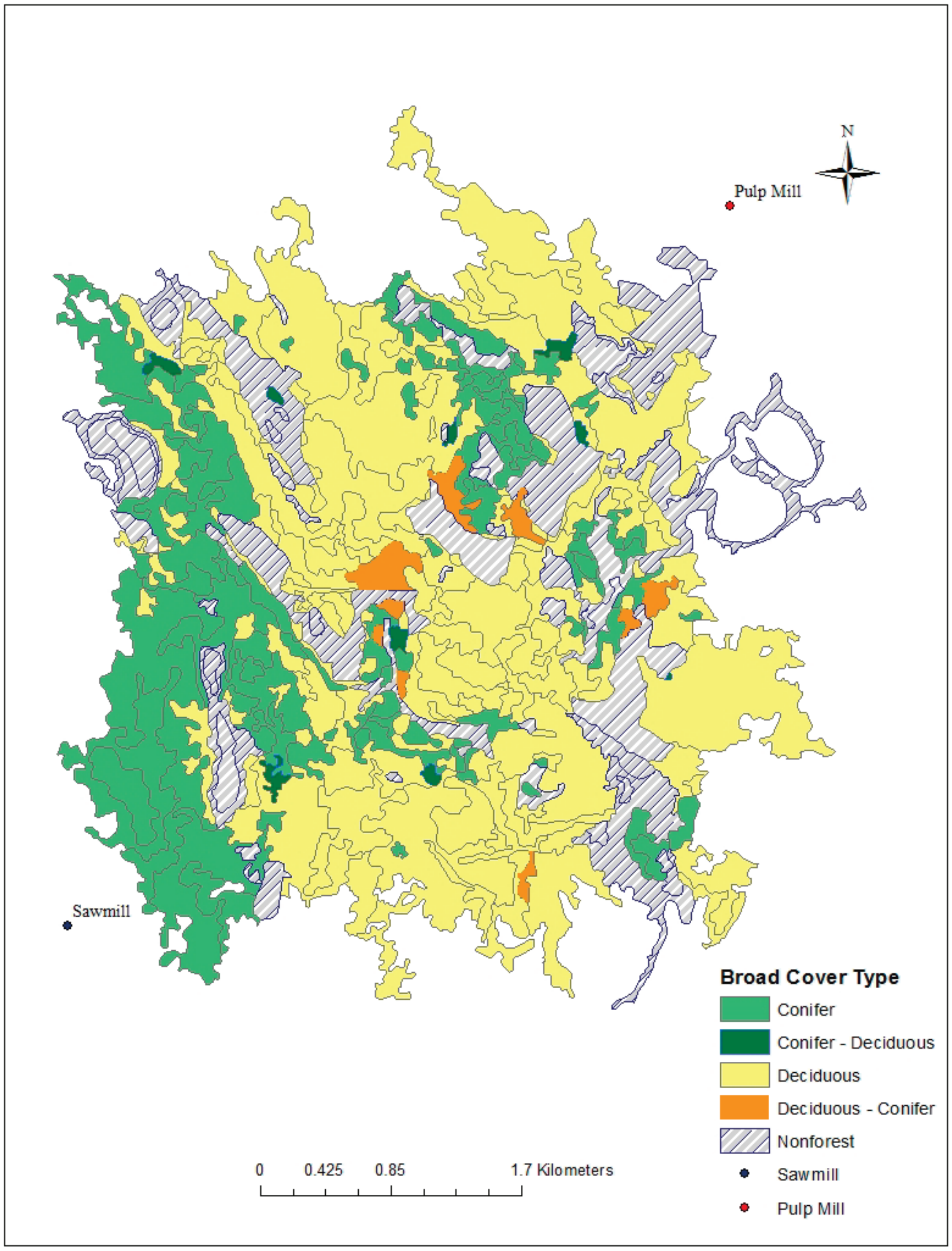


Figure 3.6: Cover type of study area

Table 3.4: Broad Cover Type Distribution

Broad Cover Type	**Area (ha)**
Conifer	417.90
Conifer - Deciduous	11.96
Deciduous	891.50
Deciduous - Conifer	29.14
Non-Forest Area	374.77

The model developed here is intended to represent a timber harvesting schedule to provide wood flow and a reserve area. Two processing facilities are represented as a sawmill and a pulp mill. The model run consists of 500 generations and each generation includes 100 harvest plans. Initial targets for the genetic algorithm model were based on the results of the mixed integer programming model. In this volume estimates come from the forest inventory file. No yield curve was used, the data stand volume was considered to be constant over the planning horizon of four time periods.

3.3.1 Input Data and Financial Parameters

Various factors limit the supply of timber in different timber supply areas. For instance, the productivity of the land may be the determining factor in one area, while the age of existing stands may be important in another. Thus, specific management activities, such as transportation, forest product manufacturing, and distribution, may influence the short or long-term timber supply in some areas more than in others and this affect the way forest grows. The age-class distribution is shown in Table 3.5, and each stand has a reserve value that is related to stand age (Fig. 3.7). The reserve quality index is hypothetical, which meant to represent that an older forest has a higher value for conservation. The area weighted average age of the study area is 67.5 years. The distribution of the stand age of the study area is shown in Figure 3.8. The discount rate is used to take into account the time value of money and the risks or uncertainty of anticipated future cash flows of forest potential. In this paper, the discount rate was set at 5 %. The mill-gate value of timber for both conifer and deciduous species was set \$100/m^3. The harvesting cost was set at \$3000/ha and haul cost was given as \$0.0273/m^3/km.

Table 3.5: Age Distribution

Age (years)	**Area (ha)**
50	4.48
60	15.89
65	61.48
70	453.19
80	14.87
85	255.05
90	247.26
100	3.91
105	107.19
110	173.30
120	12.416
130	1.44
Total	1350.4

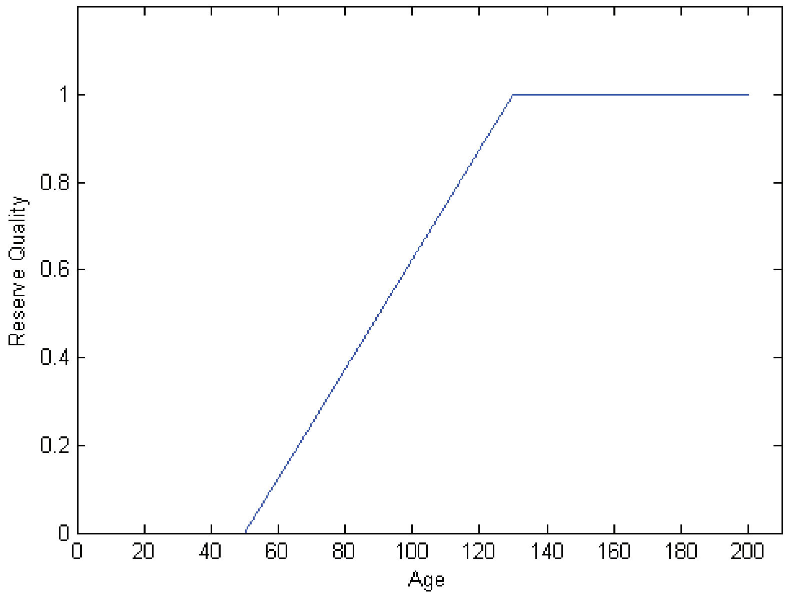


Figure 3.7: Reserve quality index

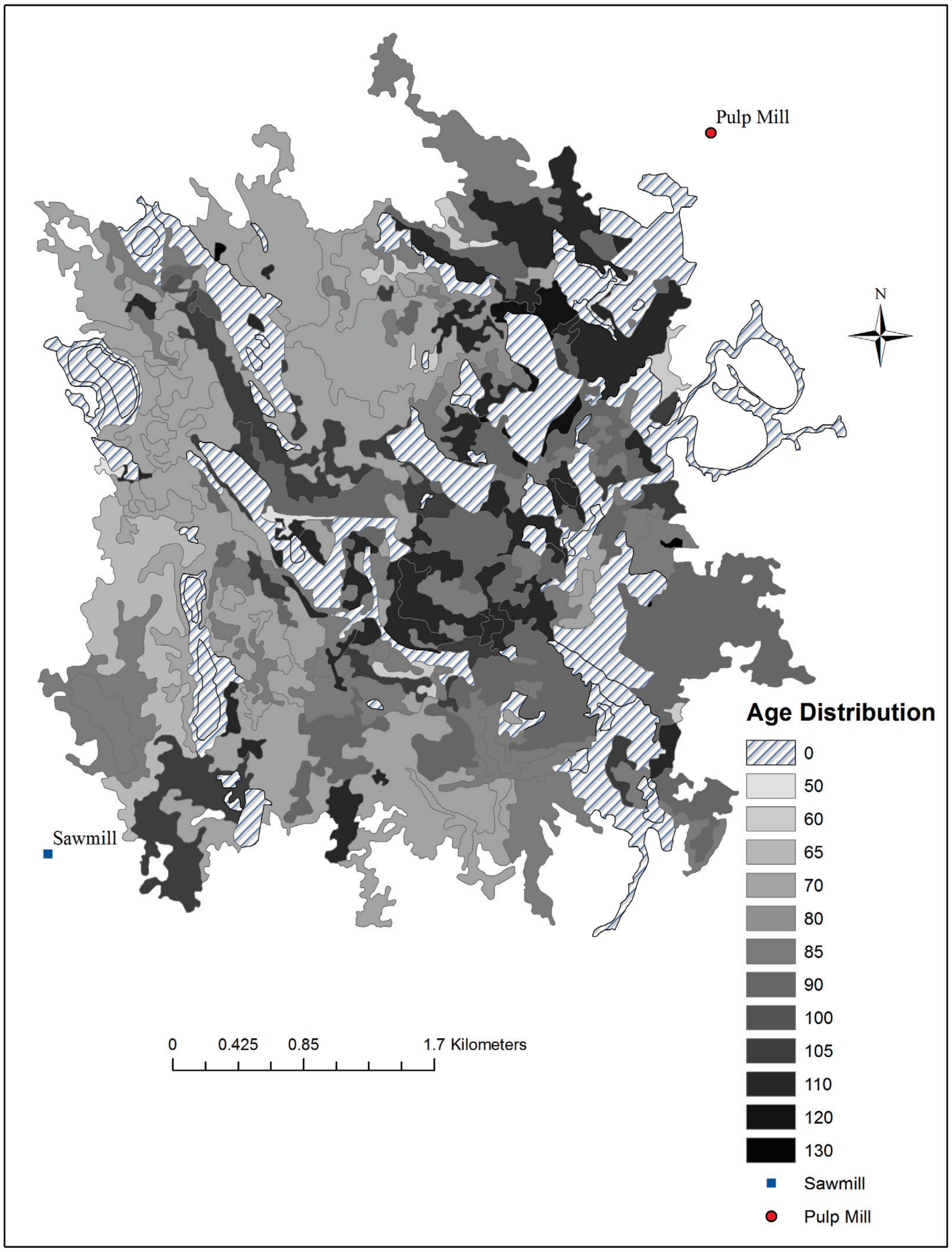


Figure 3.8: Age distribution of study area

3.4 Results & Discussion

The multi-objective genetic algorithm model was examined in this study by applying it to forest-planning problem with twelve goals. The purpose of the multi-objective optimization problem was finding an acceptable balance between all of the goals. A decision maker is implicitly considering trade-offs between conflicting goals. The genetic algorithm model rans 500 generations using 100 harvest plans as the population. The processing time for model run with penalty weights takes less than seven minutes. Each member represented a harvest schedule, and this harvest schedule was allocated by the model to the harvesting zone and the reserve zone. A near optimal harvesting schedule was selected according to the lowest penalty of a member in the 500 generations. The penalty function included twelve objectives by summing the individual objective penalty values together.

The manager is able to decide goals weights according to the purposes of forest land use. The determination of weights in the penalty function (or objective function in a goal programming framework) may be one of the most difficult tasks in this kind of problem. According to Gass (1987), it is the analyst's task to develop a suitable weighting procedure to capture the interplay between the goals of a decision makers. The weights themselves do not have intuitive meanings or interpretations.

There is no one optimal solution since different decision makers would be willing to accept different sets of trade-offs (Hotvedt, 1983). An analyst presents a decision maker with results of a series of runs in order to find a solution acceptable to the decision maker. The purpose of changing penalty weights is to reach a combination of achieved goals. In determining the penalty weights, I am playing the role of both decision makers and analyst. The analyst presents the decision maker with the results of a number of runs with different penalty weights.

The procedure I used to determine weights is described below.

Set 1 : I started by setting targets for each of the goals (Table 3.6). The net present value goal was set to \$2.00 x 10^7. The purpose of setting a higher net present value goal was to challenge the model to reach a higher NPV as much as possible. Initially the penalty for under-achievement of the net present value goal was set to 1 penalty unit per dollars (pu/\$). In the first run, the penalties for all other goals were set to zero, which makes this first run essentially a net present value maximization problem. The results of this run are shown in Table 3.6, which shows that the goal for net present value was satisfied; the other goals have unsatisfactory levels. The summary results presented in Table 3.7 are the results from the harvest schedule with the minimum penalty value across all generations. The same is true for the summary tables presented for all the sets of runs discussed here.

Set 2 : Set 2 is used to find penalty weights for periodic conifer harvests that would result in an acceptable balance between net present value and flow of conifer harvest volume. As shown in Tables 3.8 and 3.9, the model is able to achieve the target net present value and periodic coniferous harvest volume flows when the penalties for under and over-achievement of target harvest volumes are set to 1 pu/m^3. However, Table 3.9 also shows that results for the penalty weights are set to 10, 100 and 1,000 pu/m^3 because I found that the penalty needed to be increased when I ran set 3 to determine acceptable penalty weights for deciduous volume.

Set 3 : In set 3, I hoped to systematically vary the penalty weight for deciduous volume harvest while holding the penalty weight for coniferous volume constant at 1 pu/m^3. However, this did not work even when the penalty weight for deciduous harvest volume was set at 1 pu/m^3. The strategy I took instead was to vary the penalty weight for both coniferous and deciduous species simultaneously. Table 3.11 shows the achieved goals for different harvest volume penalty weights. I decided that a penalty weight of 1,000 pu/m^3 of harvest volume represented the best balance between net present value and harvest volume goals.

Set 4 : Tables 3.12 and 3.13 represent the penalty weight and results including the inclusion of penalties for over-achievement of adjacency violations. A perfect outcome would have zero adjacency violations as was forced through using constraints in the mixed integer model in chapter 2. Over-achievement of adjacency violation was penalized at 0 because fewer violations of adjacency are desired. I settled an 1 million pu/adjacency violation as the appropriate penalty for over-achievement of adjacency violation. There were only two adjacent stands harvested in the same time period in the selected solution and the net present value was still high at $8.94 million.

Set 5 : Tables 3.14 and 3.15 show the results of set 5 which was used to determine the appropriate penalty weight for over-achievement of the minimum spanning tree cost goal. The minimum spanning tree represents the minimum length of connections between all points selected as reserve area. If the goal is to produce a single large reserve, the minimum spanning tree should be short. However, at the extreme, a minimum spanning tree cost of zero could be achieved if no stands were selected for a reserve area. I selected 1,000 pu/m as the penalty for over-achievement as the minimum spanning tree cost was low, net present value was high and harvest volumes were on target. When the penalty was increased to 10,000 pu/m, harvest volumes departed from target to a level I deemed unacceptable.

Set 6 : Tables 3.16 and 3.17 show parameters and results for the final run used to determine the penalty weight for under-achievement of the quality-weighted reserve area goal. In this study, reserve quality is assumed to vary with stand age according to Figure 3.7. I set a target for the reserve area to be 25% of the total forested area, which is equal to 337 ha. Stands that are 100 years old have a quality index of 0.625 according to Figure 3.7. Accordingly, I set the quality-weighted reserve area target to be 200 ha. I determined the best weight for the reserve value to be 100,000 pu/ha, which resulted in a quality weighted reserve areas of 176.80 ha.

Tables 3.18 and 3.19 summarize the goals, penalty weights, and results for the model selected as the best at representing decision makers' preferences in relation to specified goals. Some reduction in net present value occurred as a result of the addition of adjacency violation and reserve goals. Harvest volumes were almost exactly on target, there were zero adjacency violations and the quality-weighted reserve area was on target.

The maps in Figure 3.9 and 3.10 show the harvest schedule and reserve area from the selected plan. These show that the adjacency violation goal was well satisfied, and the reserve area seemed to be more grouped than would be seen with a random assignment. When the map of the reserve areas (Fig. 3.10) is compared to the age class map (Fig. 3.8), it can be seen that many of the oldest stands were selected for the reserve area. It is likely that the genetic algorithm found a good balance between the specified minimum spanning tree and quality-weighted reserve area goals. In order to get a more aggregated reserve area, more stands of a lower quality would need to be incorporated into the reserve.

Table 3.6: **Set 1**: The development of penalty function net present value (NPV) goal weights

	Goals	**Under-achievement**	**Over-achievement**
NPV ($)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	0	0
Dec. Harv. Vol (m^3)	2.5x10^4	0	0
Adj. Viol.	0	0	0
Min. Span. Tree (m)	0	0	0
Res. Value (ha)	200	0	0

Table 3.7: Summary results considering only NPV ($) Goal Weights

	Net Present Value Weights				
Goals	**1**	**10**	**100**	**1,000**	**10,000**
NPV	1.64x10^7	-	-	-	-
Con. Har. Vol.1	8.47x10^4	-	-	-	-
Con. Har. Vol.2	478.55	-	-	-	-
Con. Har. Vol.3	73.48	-	-	-	-
Con. Har. Vol.4	72.56	-	-	-	-
Dec. Har. Vol.1	1.26x10^5	-	-	-	-
Dec. Har. Vol.2	273.67	-	-	-	-
Dec. Har. Vol.3	29.10	-	-	-	-
Dec. Har. Vol.4	5.22	-	-	-	-
Adj. Viol.	347	-	-	-	-
Min. Span. Tree	9.56x10^3	-	-	-	-
Res. Value	26.16	-	-	-	-
Penalty	3.64x10^6	-	-	-	-

Table 3.8: **Set 2**: The development of penalty function conifer harvest volume goal weights

	Goals	**Under-achievement**	**Over-achievement**
NPV ($)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	variable	variable
Dec. Harv. Vol (m^3)	2.5x10^4	0	0
Adj. Viol.	0	0	0
Min. Span. Tree (m)	0	0	0
Res. Value (ha)	200	0	0

Table 3.9: Summary results considering NPV ($) and conifer harvest volume (m^3) goal weights

	Conifer Harvest Volume Weights				
Goals	**1**	**10**	**100**	**1,000**	**10,000**
NPV	1.15x10^6	1.60x10^7	1.10 x10^7	1.03x10^6	-
Con. Har. Vol.1	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	-
Con. Har. Vol.2	1.49x10^4	1.50x10^4	1.50x10^4	1.50x10^4	-
Con. Har. Vol.3	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	-
Con. Har. Vol.4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	-
Dec. Har. Vol.1	8.38x10^4	1.23x10^5	4.27x10^4	1.48x10^5	-
Dec. Har. Vol.2	2.23x10^4	2.03x10^3	3.82x10^4	2.85x10^3	-
Dec. Har. Vol.3	1.27x10^4	268.72	2.74x10^4	997	-
Dec. Har. Vol.4	5.29x10^3	50.64	2.13x10^4	8.46x10^2	-
Adj. Viol.	177	209	102	146	-
Min. Span. Tree	8.27x10^3	1.12x10^4	1.47x10^4	9.09x10^3	-
Res. Value	123.76	16.90	86.11	174.18	-
Penalty	1.27x10^7	4.74 x10^6	7.38x10^6	9.87x10^6	-

Table 3.10: **Set 3**: The development of penalty function deciduous harvest volume goal weights

	Goals	**Under-achievement**	**Over-achievement**
NPV ($)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	variable	variable
Dec. Harv. Vol (m^3)	2.5x10^4	variable	variable
Adj. Viol.	0	0	0
Min. Span. Tree (m)	0	0	0
Res. Value (ha)	200	0	0

Table 3.11: Summary results considering NPV ($), conifer and deciduous harvest volume (m^3) goal weights

	Deciduous Harvest Volume Weights				
Goals	**1**	**10**	**100**	**1,000**	**10,000**
NPV	1.12x10^7	1.56x10^7	8.62x10^6	8.83x10^6	-
Con. Har. Vol.1	7.25x10^4	6.83x10^4	1.50x10^4	1.50x10^4	-
Con. Har. Vol.2	1.85x10^4	1.50x10^4	1.50x10^4	1.50x10^4	-
Con. Har. Vol.3	1.50x10^4	1.85x10^4	1.50x10^4	1.50x10^4	-
Con. Har. Vol.4	379.01	260	1.50x10^4	1.50x10^4	-
Dec. Har. Vol.1	1.24x10^5	1.00^5	2.51x10^4	2.50x10^4	-
Dec. Har. Vol.2	683.39	2.46x10^4	2.52x10^4	2.50x10^4	-
Dec. Har. Vol.3	1.10x10^4	578.49	2.51x10^4	2.50x10^4	-
Dec. Har. Vol.4	1.06x10^4	14.72	2.51x10^4	2.50x10^4	-
Adj. Viol.	190	47	72	63	-
Min. Span. Tree	9.41x10^3	8.02x10^3	1.92x10^4	1.82x10^4	-
Res. Value	126.15	16.7	123.02	111.57	-
Penalty	4.02x10^6	6.40x10^6	1.39x10^6	1.12x10^7	-

Table 3.12: **Set 4**: The development of penalty function adjacency violation goal weights

	Goals	**Under-achievement**	**Over-achievement**
NPV (\$)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	1,000	1,000
Dec. Harv. Vol (m^3)	2.5x10^4	1,000	1,000
Adj. Viol.	0	0	variable
Min. Span. Tree (m)	0	0	0
Res. Value (ha)	200	0	0

Table 3.13: Summary results considering NPV (\$), conifer and deciduous harvest volume (m^3) and adjacency violation goal weights

	Adjacency Violation Weights				
Goals	**100**	**1,000**	**10,000**	**100,000**	**1,000,000**
NPV	8.86x10^6	8.82x10^6	8.88x10^6	8.85x10^6	8.84x10^6
Con. Har. Vol.1	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4
Con. Har. Vol.2	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4
Con. Har. Vol.3	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4
Con. Har. Vol.4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4
Dec. Har. Vol.1	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Dec. Har. Vol.2	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Dec. Har. Vol.3	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Dec. Har. Vol.4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Adj. Viol.	69	77	53	21	2
Min. Span. Tree	1.90x10^4	1.72x10^4	1.80x10^4	1.90x10^4	1.54x10^4
Res. Value	105.35	115.53	118.11	119.32	126.89
Penalty	1.12x10^7	1.13x10^7	1.17x10^7	1.34x10^7	1.22x10^7

Table 3.14: **Set 5**: The development of penalty function minimum spanning tree goal weights

	Goals	**Under-achievement**	**Over-achievement**
NPV ($)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	1,000	1,000
Dec. Harv. Vol (m^3)	2.5x10^4	1,000	1,000
Adj. Viol.	0	0	1,000,000
Min. Span. Tree (m)	0	0	variable
Res. Value (ha)	200	0	0

Table 3.15: Summary results considering NPV ($), conifer and deciduous harvest volume (m^3) and adjacency violation and minimum spanning tree (m) goal weights

	Minimum Spanning Tree Weights				
Goals	**1**	**10**	**100**	**1,000**	**10,000**
NPV	8.81x10^6	8.88x10^6	9.09x10^6	8.77x10^6	8.63x10^6
Con. Har. Vol.1	1.47x10^4	1.50x10^4	1.50x10^4	1.51x10^4	1.67x10^4
Con. Har. Vol.2	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	2.66x10^4
Con. Har. Vol.3	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.65x10^4
Con. Har. Vol.4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.48x10^4
Dec. Har. Vol.1	2.44x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.51x10^4
Dec. Har. Vol.2	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.61x10^4
Dec. Har. Vol.3	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.54x10^4
Dec. Har. Vol.4	2.48x10^4	2.50x10^4	2.50x10^4	2.49x10^4	2.50x10^4
Adj. Viol.	6	2	3	2	8
Min. Span. Tree	1.83x10^4	1.95x10^4	1.82x10^4	1.18x10^4	4.12x10^3
Res. Value	102.41	125.75	145.11	132.62	86.29
Penalty	8.01x10^6	3.31x10^6	9.07x10^6	2.73x10^7	7.21x10^7

Table 3.16: **Set 6**: The development of penalty function reserve value goal weights

	Goals	**Under-achievement**	**Over-achievement**
NPV ($)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	1,000	1,000
Dec. Harv. Vol (m^3)	2.5x10^4	1,000	1,000
Adj. Viol.	0	0	1,000,000
Min. Span. Tree (m)	0	0	1,000
Res. Value (ha)	200	variable	0

Table 3.17: Summary results considering NPV ($), conifer and deciduous harvest volume (m^3) and adjacency violation and minimum spanning tree (m) and reserve value (ha) goal weights

	Reserve Value Weights				
Goals	**10**	**100**	**1,000**	**10,000**	**100,000**
NPV	8.90x10^6	8.84x10^6	8.96x10^6	8.98x10^6	9.01x10^6
Con. Har. Vol.1	1.50x10^4	1.50x10^4	1.52x10^4	1.49x10^4	1.49x10^4
Con. Har. Vol.2	1.51x10^4	1.58x10^4	1.49x10^4	1.51x10^4	1.50x10^4
Con. Har. Vol.3	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4	1.50x10^4
Con. Har. Vol.4	1.50x10^4	1.50x10^4	1.49x10^4	1.50x10^4	1.50x10^4
Dec. Har. Vol.1	2.50x10^4	2.49x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Dec. Har. Vol.2	2.51x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Dec. Har. Vol.3	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4	2.50x10^4
Dec. Har. Vol.4	2.52x10^4	2.50x10^4	2.52x10^4	2.50x10^4	2.50x10^4
Adj. Viol.	3	2	2	2	0
Min. Span. Tree	1.31x10^4	1.35x10^4	1.35x10^4	1.49x10^4	1.39x10^4
Res. Value	117.78	130.07	126.08	124.51	176.80
Penalty	1.79x10^7	2.02x10^7	1.72x10^7	1.81x10^7	3.18x10^7

Table 3.18: The final penalty function weights of each goal

	Goals	**Under-achievement**	**Over-achievement**
NPV ($)	2.0x10^7	1	0
Con. Harv. Vol (m^3)	1.5x10^4	1,000	1,000
Dec. Harv. Vol (m^3)	2.5x10^4	1,000	1,000
Adj. Viol.	0	0	1,000,000
Min. Span. Tree	0	0	1,000
Res. Value (ha)	200	100,000	0

Table 3.19: Summary table for selected plan when all goals are penalized

Penalty	3.184 x 10^7
Net Present Value ($)	8.88 x 10^6
Harvest Volume of Conifer (m^3)	
Period 1	1.491 x 10^4
Period 2	1.504 x 10^4
Period 3	1.502 x 10^4
Period 4	1.503 x 10^4
Total Volume	6.000 x 10^4
Harvest Volume of Deciduous (m^3)	
Period 1	2.503 x 10^4
Period 2	2.501 x 10^4
Period 3	2.502 x 10^4
Period 4	2.502 x 10^4
Total Volume	1.008 x 10^5
Adjacency Violation	0
Min. Span. Tree Weight (m)	1.364 x 10^4
Reserve Value (ha)	176.803
Reserve Area (ha)	362.340
Scheduled Stands	231
Harvesting Zone	179
Reserve Zone	52

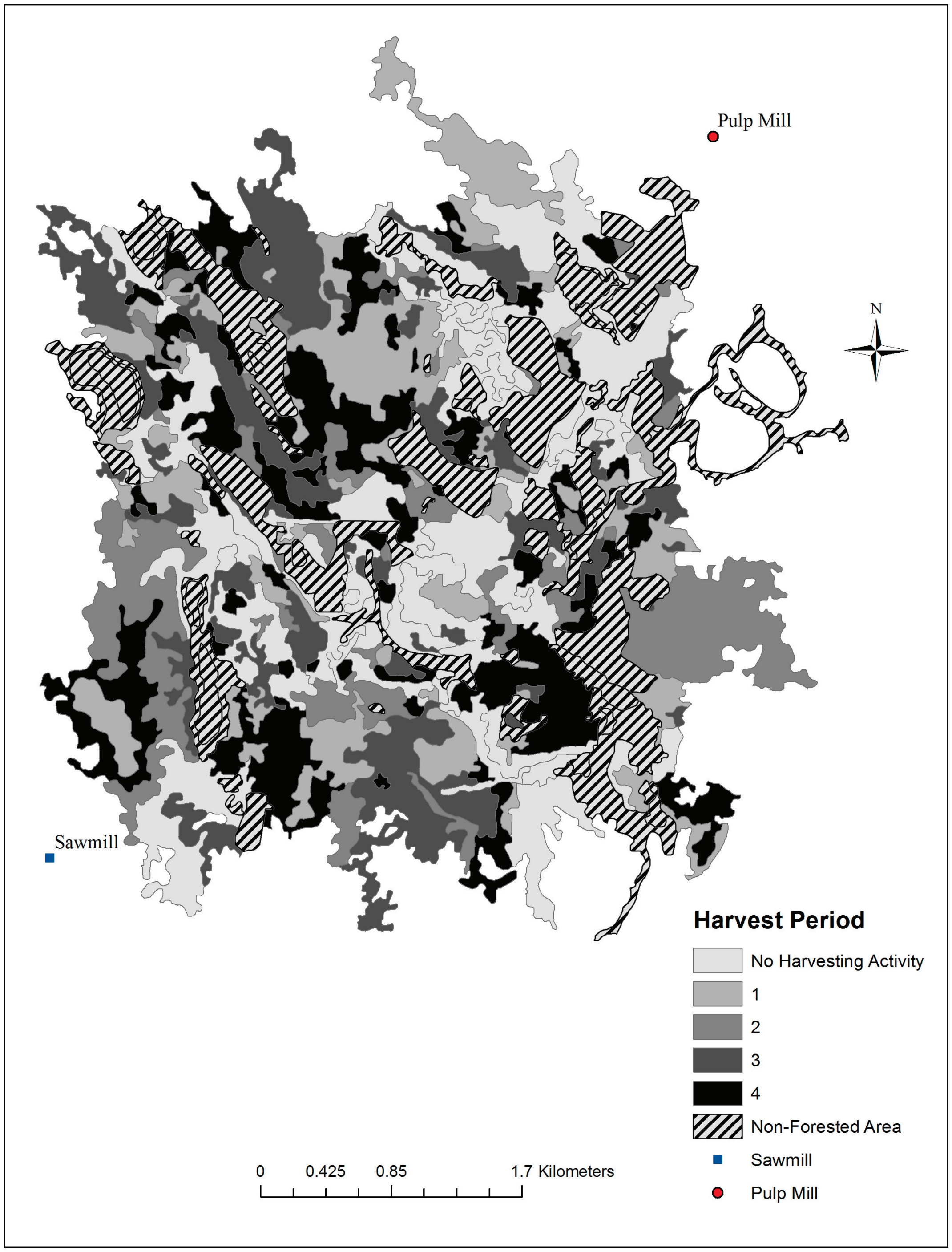


Figure 3.9: Harvesting periods for selected solution of study area

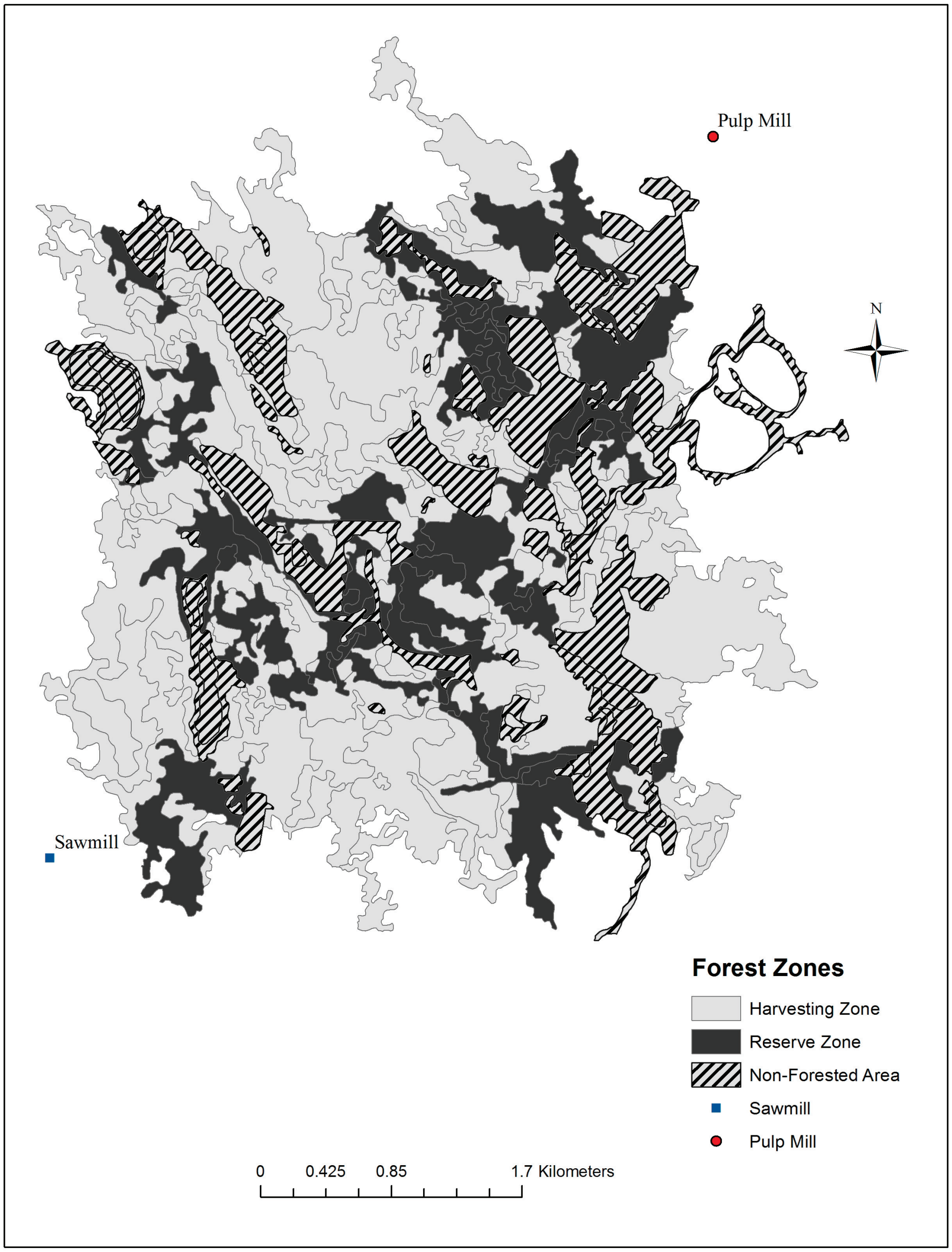


Figure 3.10: Forest zones for selected solution of Study Area

The Figure 3.11 shows the evolution of minimum penalty in each generation. The graph was created by a minimum penalty of members in the generation. By generation 350, the improvement in penalty function was small. The results represent net present value at $ 8.88 million which is still good when compared to the results of mixed integer goal programming model (Fig 3.12). Out of 287 stands, 179 are scheduled for the harvesting zone while 52 stands are for the reserve zone. Total harvest volumes for coniferous and deciduous species are represented in Figure 3.13, which shows the fluctuation of harvest volumes over all generation that tends to improve and stabilize with generation over time. Figure 3.14 represents how adjacency violation changes among 500 generations.

The area weighted stand age of the reserve zone was 95.14 years. The reserve zone has 356.3418 ha and 176.80 reserve quality value (Fig. 3.15). Reserve zone consists of 161.68 ha conifer species and 200.66 ha deciduous species. The management goal of the reserve zone is to group those stands together. In this study, a single large stand preferred over several small stands. It is assigned that reserve stands be close together. The final goal finds the minimum distance between stands by using minimum spanning tree function that aggregate the reserve areas by minimizing the cost of the minimum spanning tree. The model used in this study is useful in planning, communication or transportation networks where the objective function is to provide some connecting route between nodes at the lowest possible cost. The Delaunay triangulation method created every possible triangle but chose triangles which were not long and have large angles. Delaunay triangulation helps to construct a minimum spanning tree. All nodes in a minimum spanning tree are edges of neighborhood graph. Figure 3.1 shows how minimum spanning tree cost is found according to the results of the model for selected harvest schedule.

Different mutation rates and numbers of population were tested in this study. Box plots in 3.17 display differences of minimum penalty values at the y-axis for different population sizes and mutation rates without making any assumptions about the underlying statistical distribution. These graphs depict groups of minimum penalties across all generations through 50 draws. When compared to the medians of the each combination, the lowest penalty value is resulting if population size is 200 and mutation rate is 0.010%. One hundred and two hundreds population size resulted slightly better solution than population size of 50. A higher mutation rate has fluctuation of minimum penalty of each generation, which provides more variation. The model with a small mutation rate and larger population size ran longer to find a near optimal solution. For this problem, decreasing population size helps speed up processing time and increasing the mutation rate might be good for achieving diversity.

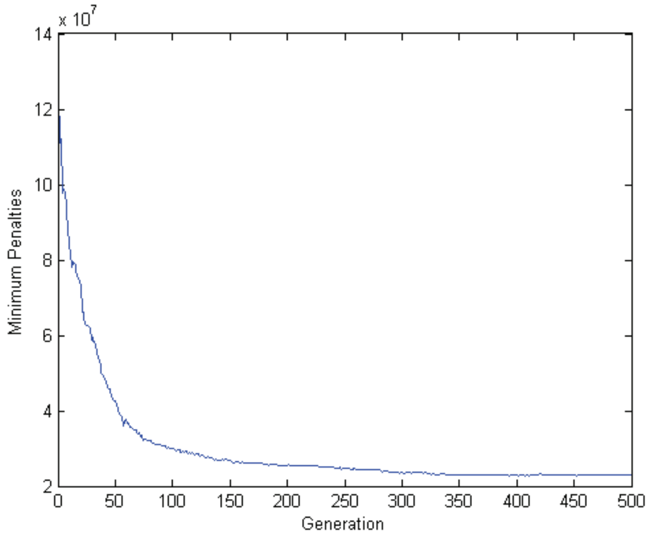


Figure 3.11: Minimum penalties of 500 generations for selected harvest schedules

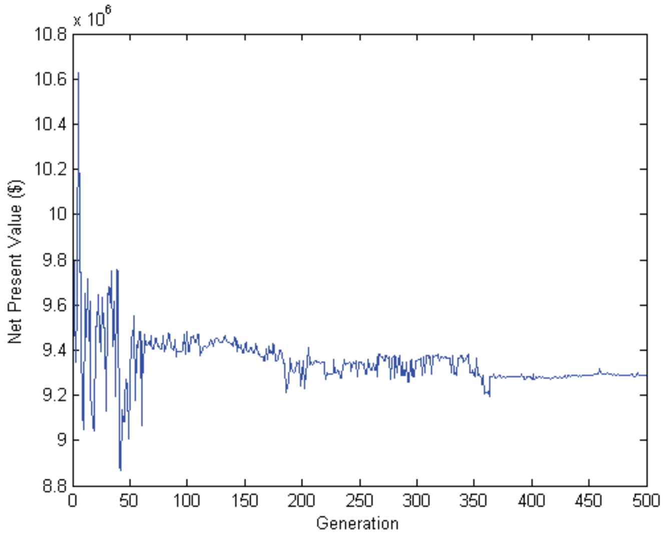
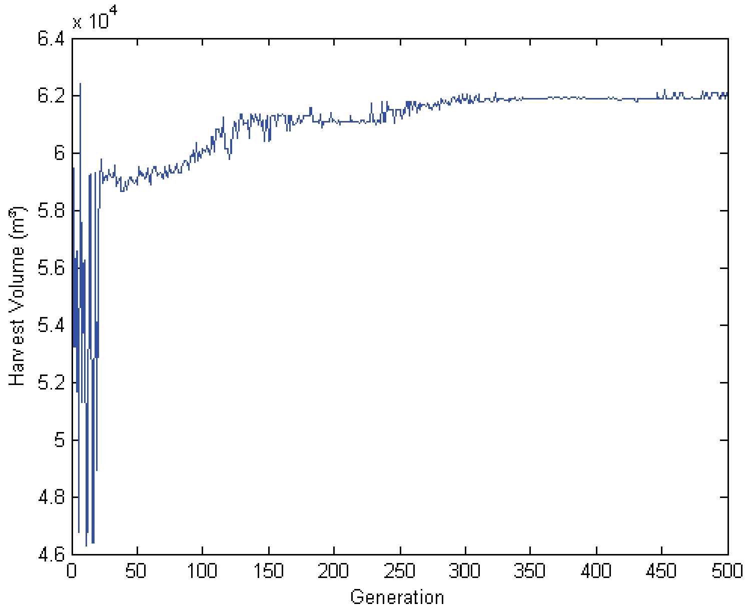
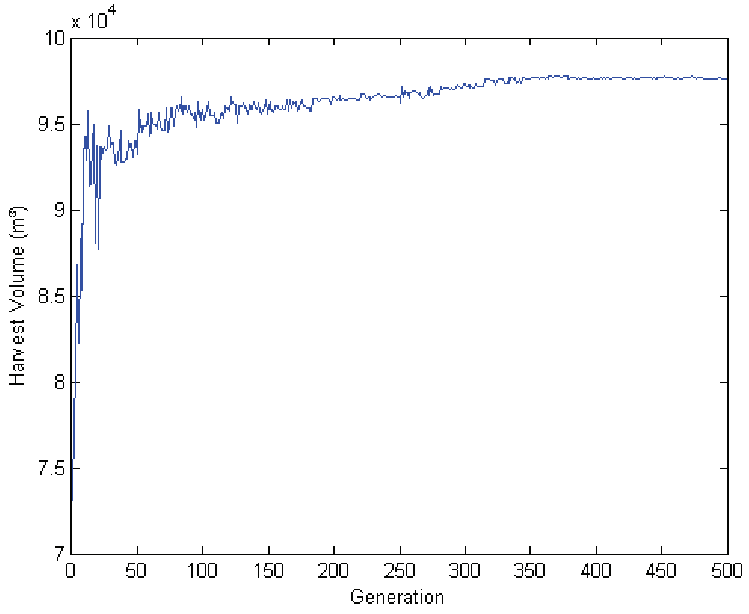


Figure 3.12: Net present value for selected harvest schedules of each generation



(a) Total harvest volume of conifer



(b) Total harvest volume of deciduous

Figure 3.13: Harvest volume for selected harvest schedules of each generation

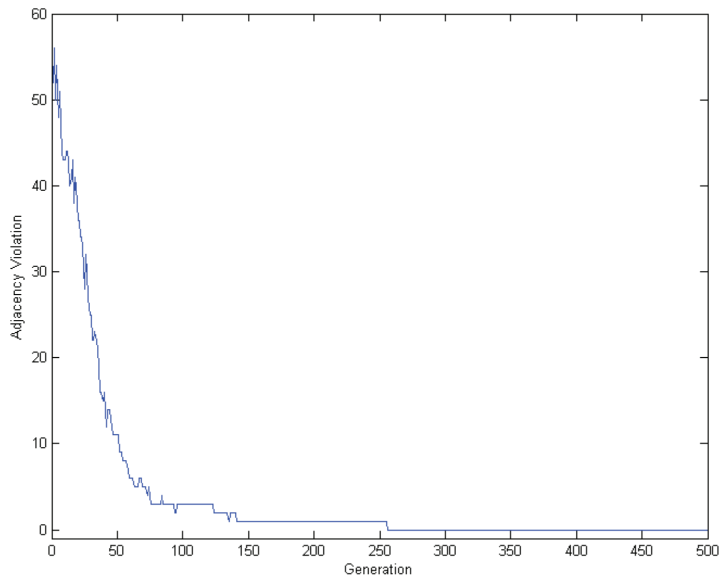


Figure 3.14: Distribution of adjacency violations for selected harvest schedules of each generation

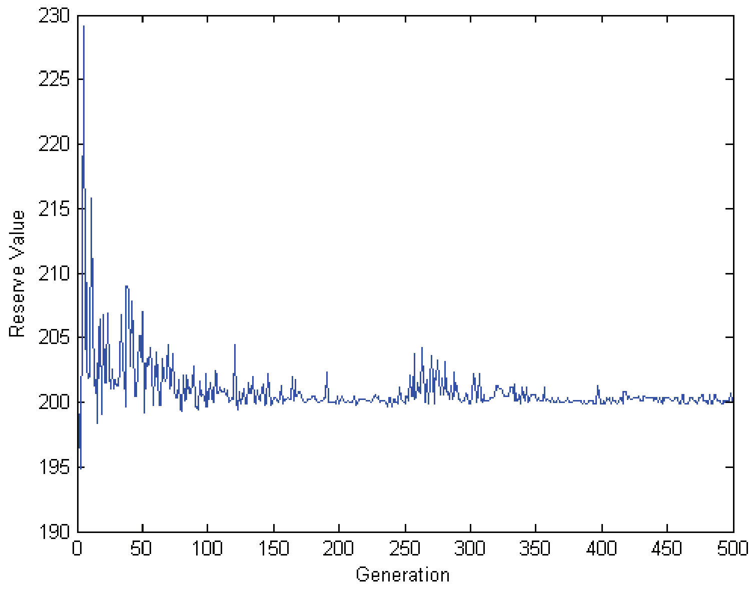


Figure 3.15: Reserve value for selected harvest schedules of each generation

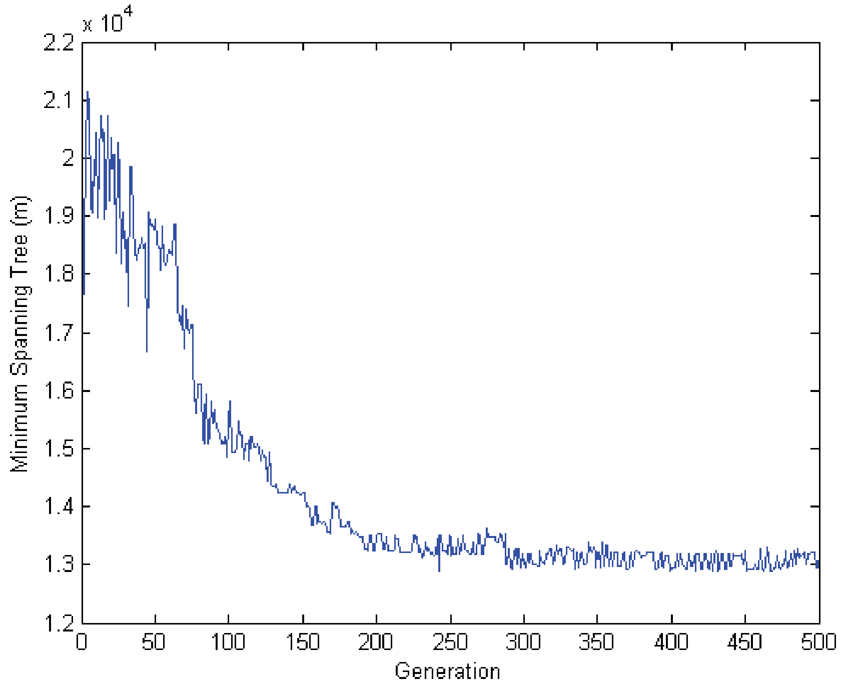


Figure 3.16: Minimum spanning tree for selected harvest schedules of each generation

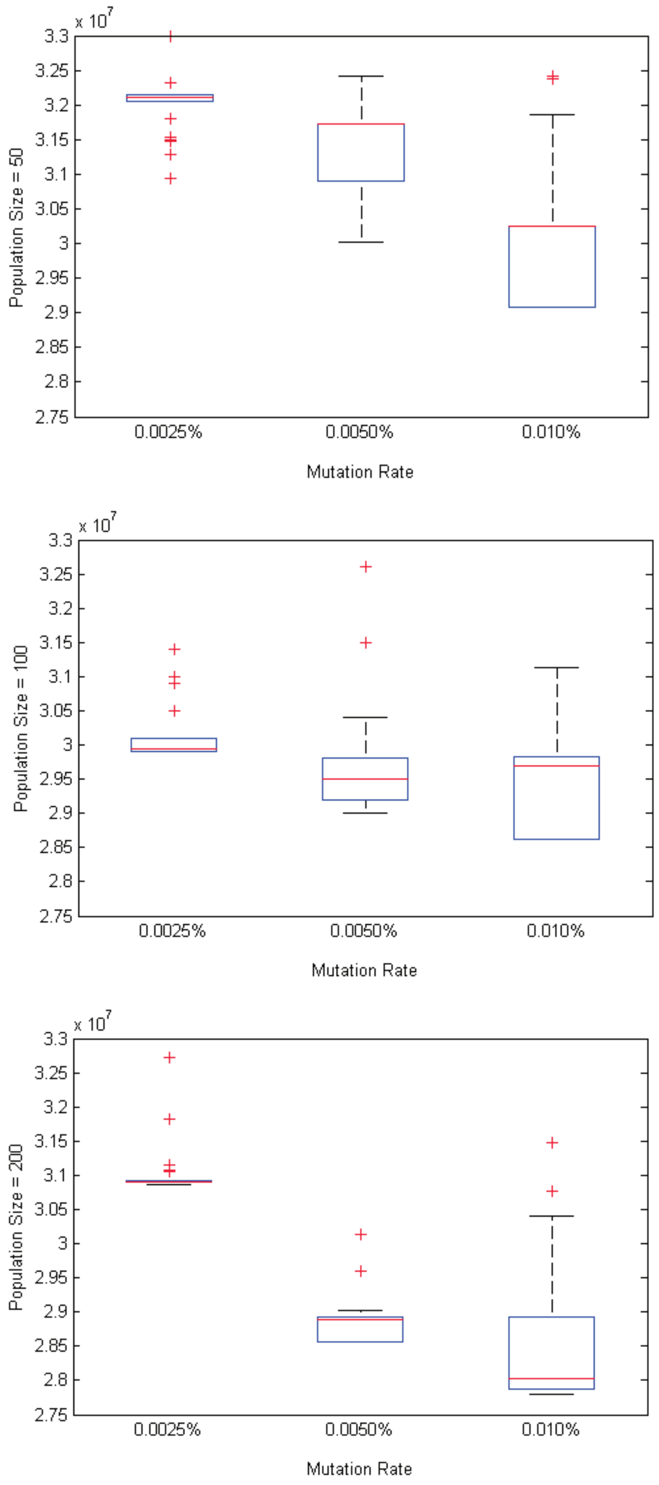


Figure 3.17: Minimum penalty graphs with different population size and mutation rate

3.5 Conclusion

Decision-making in forest planning is a multidimensional decision problem, concerned with multiple sustainable use of the forests. While performing searches in a large-scale problem, the genetic algorithm models offer significant benefits over optimization techniques such as mixed integer programming and goal programming. Genetic algorithm models are good at taking large, potentially huge search spaces and navigating them to look for optimal combinations of the solutions that might otherwise take a lifetime to find (Mangano, 1995).

When problems become bigger, it is difficult to solve with mixed integer programming model. Moreover, some formulation of constraints are not able to solve with mixed integer programming framework. Given the computational complexity, finding a goal programming model for a feasible solution is essentially as hard as actually finding out the optimal solution. Because of the complexity of the programming and difficulties applying adjacency violation and minimum spanning tree aspects, the linear programming approaches applied in chapter 2 were converted to the genetic algorithm model. This problem is well suited to finding an acceptable solution through a heuristic optimization procedure, developed here as a genetic algorithm model. The optimization built here started with an initial population of harvest schedules. These were generated by randomly assigning a harvest period to each stand in the forest for each member of the initial population. The penalty function was used to calculate a penalty value for each member population. The penalty value was the most important part of the decision-making process.

Using a small mutation rate should have a tendency to preserve a good current solution while using high crossover and mutation rates should suggest new solutions. The implementation of a genetic algorithm should maintain balance between these aspects (Lu and Eriksson, 2000). The selection rate in this model was based on individual fitness. Modifying mutation rate affects the penalties standard deviation. The population with the smallest mutation rate had smaller standard deviations. The higher a mutation rate in this model increases the probability of finding different penalty ranges.

Selection is a mechanism in the genetic algorithm that provides some members of the population the opportunity to be parents and pass down their genetic information. The fitness function designed in this model calculated each individual's fitness value through a comparison of that individual's penalty score with those of the rest of the population to find the highest fitness value (Shiffman et al., 2012). The main idea is to search for contiguous sets of stands that each represent a potential adjacency violation. The adjacency violation constraint in the penalty function tries to reach minimum violation occurrence. All of the stands in the management area are contributing to the adjacency constraint violations and are added to the adjacency matrix. The adjacency violation goal in this model was set as 0 and compared differences by changing over achievement of the violation. Grouping forest stands for the harvesting zone with a minimum spanning tree may apply to the harvesting area to reduce the movements of the harvester, forwarders and staff. It may also help to the adjacency restriction, thus maximum land use might be achieved.

This model achieved a solution that allows calculating the area to be harvested during each period with profits as high as possible. It ensured a balanced reserve zone and harvesting zone distribution by the end of the planning horizon, which satisfies the wishes of the decision-maker. I was able to find a satisfactory solution with added reserve quality, group to the reserve zone and adjacency violation count thanks to the flexibility of the genetic algorithm.

Bibliography

Diamond, J. M., 1975. The island dilemma: Lessons of modern biogeographic studies for the design of natural reserves. Biol. Conser. 7 (2), 129–146.

Dykstra, D., 1984. Mathematical programming for natural resource management. McGraw-Hill Series in Forest Resources. McGraw-Hill, New York. 318p.

Fotakis, D. G., Sidiropoulos, E., Myronidis, D., Ioannou, K., 2012. Spatial genetic algorithm for multi-objective forest planning. Forest Policy Econ. 21, 12–19.

Gass, S. I., 1987. The setting of weights in linear goal-programming problems. Computers & operations research 14 (3), 227–229.

Hotvedt, J. E., 1983. Application of linear goal programming to forest harvest scheduling. Southern Journal of Agricultural Economics 15 (1), 103–108.

Kangas, A., Kangas, J., Kurttila, M., 2008. Decision support for forest management. Vol. 16. Dordrecht, Springer Series on Managing Forest Ecosystems.

Kundur, V., 2005. Prim's algortihm [Matlab Code] Available at: http://www.math.washington.edu/greenbau/math_381/programs/prim.html [Accessed May 10, 2013].

Lu, F., Eriksson, L. O., 2000. Formation of harvest units with genetic algorithms. Forest Ecol. Man. 130 (1), 57–67.

Maene, C., 2011. Adjacent & neighboring polygons ARCGIS tool box. Available at: http://resources.arcgis. com/gallery/file/geoprocessing [Accessed February 10, 2013].

Mangano, S., 1995. Genetic algorithms solve seemingly intractable problems. Computer Design 34 (5), 70–91.

MATLAB, 2010. version 7.10.0 (R2010a). The MathWorks Inc., Natick, Massachusetts.

Mullen, D. S., 2000. An optimal approach and algorithms for limiting forest opening size in spatially constrained timber harvest scheduling models. USDA. Forest Serv. Gen. Tech. Rep. Nc. 1, 74–80.

Prim, R. C., 1957. Shortest connection networks and some generalizations. Bell System Technical Journal 36 (6), 1389–1401.

Shiffman, D., Fry, S., Marsh, Z., 2012. The Nature of Code. Theoklesia, LLC.

Smaltschinski, T., Seeling, U., Becker, G., 2012. Clustering forest harvest stands on spatial networks for optimised harvest scheduling. Annals of Forest Science 69 (5), 651–657.

Chapter 4

Conclusion

This thesis presented a study related to a forest planning model for timber harvesting and dividing forest land into zone types based on objectives. The first chapter focused on an introduction to the general approach to solving spatial forest planning problems which includes linear programming and heuristic methods to solve complicated problems. The second chapter represented mixed integer programming and mixed goal programming model formulations and solutions of sample spatial forest planning problem. The third chapter examined the near optimal solution by using a genetic algorithm model that creates harvesting and reserve zones by gathering information from a mixed integer programming model and the idea is adapted from a mixed goal programming model.

The main objective of this thesis is the development of a genetic algorithm model to minimize sum of the weighted deviations of net present value, the harvesting goal for the operational planning on a periodic level, count of adjacency violations, aggregated reserve stands with minimum spanning tree method and target quality weighted reserve areas. The genetic algorithm was successfully applied to solve a harvest scheduling problem for a forest area located in Alberta. The problem involved 287 forest stands to be cut over four periods. Forests have important functions such as recreation, timber production and nature conservation. Allocating the zones of a forest landscape is an important and contentious procedure. The location and suitability of land has to be assigned land use objectives to obtain the highest value from zoning. Once zones are allocated, it is difficult to move from a production/intensive use zone to a reserve zone and/or one with conservation objectives (Bos, 1993). The difference between mixed integer programming and genetic algorithm is exploring different formulations of penalty/objective function. There is no linear requirement for penalty function in genetic algorithm. The key idea of the mixed integer programming was to find out the maximum timber harvesting volume and net present value by considering the area is not divided into other zones. The penalty function designed for deviations of the target values (adjacency violation, net present value and harvest volume) to penalize. Reserve areas are elaborated by minimum spanning tree algorithm, which helped to create the adjacency matrix and to achieve minimum cost weight simplifies to manage planning units.

The main advantage of genetic algorithm (GA) over the mixed integer goal programming (MIGP) approach is that it is easy to modify the model with a complex algorithm such as adjacency violation and minimum spanning tree function. The genetic algorithm model is well suited to the solution of forest harvesting scheduling problems. GA models are efficient and can be easily run with a range of objective functions. The developed models MIGP model can be successfully applied and will ensure an optimum harvesting schedule over different time periods. GA model can find a near optimal solution while providing forest allocation. The mutation parameter in the genetic algorithm should be decided carefully and also be improved the role in the model because higher mutation rates force to adopt environment. Genetic algorithms also produce solutions that work within the test environment and real world by obtaining other solutions with different weight for the decision makers. Goals in the genetic algorithm model can be easily modified for decision makers demands.

In this study, I played the roles of both decision maker and analyst. The solution produced by the genetic algorithm model would be better representation of harvest schedule if a real decision maker had participated in the decision making process. The genetic algorithm model developed here is promising for solving spatial forest planning problems although the results were hypothetical.

For further studies the genetic algorithm model can be applied to larger forested areas with realistic data and intended to be better solution especially for the reserve areas. A systematic approach to determining appropriate penalty weights can help to identify decision makers preference. Minimum spanning tree algorithm is only applied to reserve areas to aggregate these areas. The minimum spanning tree approach might also be used to place harvest blocks close to each other to minimize equipment movements costs. The fictional mills represented here would help to elaborate the value chain optimization in an analysis tool to assess the sources of the competitive advantage, which examines all activities an organization performs and how each activity interaction is necessary to analyze the source of competitive advantage. Forest resource management organizations optimize the value of activities and maintain a sustainable competitive advantage (Wang et al., 2012). The model is expected to give a realistic solution when yield curve is applied. Moreover, the penalty function can incorporate other variables such as road building and access development for area. The model can be formulated differently and applied for the real forest management problems tend to set different weights from goals.

In planning and decision making processes, the roles of analyst and decision maker are more appropriately prompted by considering multiple objectives. An analyst generates alternative solutions, and a decision maker uses the solutions. More realistic models of a problem are elaborated if many objectives are considered. For this study, I gained knowledge about the process of genetic algorithms and building a model. I found out that genetic algorithms are not difficult to apply to spatial forest planning problems and are able to find a near optimal solution for a harvest schedule and simultaneous zoning approach.

Bibliography

Bos, J., 1993. Zoning in forest management: a quadratic assignment problem solved by simulated annealing. J. Environ. Manage. 37 (2), 127–145.

Wang, J., Geng, Y., Pan, K., 2012. The study on management and development of the value chain construction of state-owned forest resources operational organization. In: Soft Computing in Information Communication Technology. Springer, pp. 531–536.

BIBLIOGRAPHY

Bibliography

Anderson, J. A., Armstrong, G. W., Luckert, M. K., Adamowicz, W. L., 2012. Optimal zoning of forested land considering the contribution of exotic plantations. Mathematical & Computational Forestry & Natural Resource Sciences 4 (2).

Bettinger, P., Boston, K., Siry, J., Grebner, D. L., 2010. Forest Management and Planning. Academic Press, San Diego, CA.

Bettinger, P., Graetz, D., Boston, K., Sessions, J., Chung, W., 2002. Eight heuristic planning techniques applied to three increasingly difficult wildlife planning problems. Silva Fenn. 36 (2), 561–584.

Bettinger, P., Johnson, S. J., Debora, L., Johnson, K. N., 2003. Spatial forest plan development with ecological and economic goals. Ecol. Model. 169 (2), 215–236.

Bettinger, P., Sessions, J., 2003. Spatial forest planning: To adopt, or not to adopt? J. Forest. 101 (2), 24–29.

Bettinger, P., Sessions, J., Boston, K., 1997. Using tabu search to schedule timber harvests subject to spatial wildlife goals for big game. Ecol. Model. 94 (2), 111–123.

Bettinger, P., Zhu, J., 2006. A new heuristic method for solving spatially constrained forest planning problems based on mitigation of infeasibilities radiating outward from a forced choice. Silva Fenn. 40(2), 315–333.

Bos, J., 1993. Zoning in forest management: a quadratic assignment problem solved by simulated annealing. J. Environ. Manage. 37 (2), 127–145.

Boston, K., Bettinger, P., 1999. An analysis of monte carlo integer programming, simulated annealing, and tabu search heuristics for solving spatial harvest scheduling problems. Forest Sci. 45 (2), 292–301.

Boston, K., Bettinger, P., 2002. Combining tabu search and genetic algorithm heuristic techniques to solve spatial harvest scheduling problems. Forest Sci. 48 (1), 35–46.

Boyland, M., Nelson, J., Brodie, J Douglas Bunnell, F. L., 2004. Creating land allocation zones for forest management: A simulated annealing approach. Can. J. Forest Res. 34 (8), 1669–1682.

Bradley, S., Hax, A., Magnanti, T., 1977. Applied mathematical programming. Addison Wesley.

Broad, L., 1985. A mixed integer linear programming approach to forest utilisation management problems. University of Canterbury. Forestry.

Brundtland, G. H., 1987. Report of the World Commission on environment and development: "our common future". United Nations.

Buongiorno, J., Gilless, J. K., 2003. Decision Methods for Forest Resource Management. Academic Press., New York.

CCFM, 1995. Defining sustainable forest management: A Canadian approach to criteria and indicators. Canadian Council of Forest Ministers, Ottawa. 22 p. Available at: http://www.ccfm.org/english/coreproducts-criteria_in.asp.

CCFM, 2003. Defining Sustainable forest management in Canada: Criteria and Indicators. Canadian Council of Forest Ministers, Ottawa. 21 p. Available at: http://www.ccfm.org/english/coreproducts-criteria_in.asp.

Chappelle, D. E., 1966. A computer program for calculating allowable cut using the area-volume check method. Research Note PNW-44, USDA Forest Service. Pacific Northwest Forest and Range Experiment Station, Portland, Oregon, USA.

Chappelle, D. E., Sassaman, R., 1968. A computer program for scheduling allowable cut using either area or volume regulation during sequential planning periods. Research Note. Portland. USDA Forest Service, Pacific Northwest Forest and Range Experiment Station.

Chinneck, J. W., 2004. Practical optimization: a gentle introduction. Electronic document: http://www.sce.carleton.ca/faculty/chinneck/po.html.

Clutter, J. L., 1968. Max-million: a computerized forest management planning system. The University of Georgia. School of Forest Resources. Biometrics-Operations Research Section.

Cote, P., Tittler, R., Messier, C., Kneeshaw, D. D., Fall, A., Fortin, M.-J., 2010. Comparing different forest zoning options for landscape-scale management of the boreal forest: possible benefits of the triad. Forest Ecol. and Manag. 259 (3), 418–427.

Crowe, K. A., Nelson, J., 2005. An evaluation of the simulated annealing algorithm for solving the area-restricted harvest-scheduling model against optimal benchmarks. Can. J. Forest Res. 35 (10), 2500–2509.

Dantzig, G. B., 1965. Linear programming and extensions. Princeton University Press.

Diamond, J. M., 1975. The island dilemma: Lessons of modern biogeographic studies for the design of natural reserves. Biol. Conser. 7 (2), 129–146.

Dykstra, D., 1984. Mathematical programming for natural resource management. McGraw-Hill Series in Forest Resources. McGraw-Hill, New York. 318p.

ESRD, 2012. Alberta timber harvest planning and operating ground rules framework. Alberta Environment and Sustainable Resource Development. Available at: http://esrd.alberta.ca/lands-forests/forest-management /documents/TimberHarvest-OperatingGroundRules-Jun2012.pdf. [Accessed 2014-04-21].

Field, D. B., 1973. Goal programming for forest management. Forest Sci. 19 (2), 125–135.

Field, R. C., Dress, P. E., Fortson, J. C., 1980. Complementary linear and goal programming procedures for timber harvest scheduling. Forest Science 26 (1), 121–133.

Fotakis, Dimitris G; Sidiropoulos, E. M. D. I. K., 2012. Spatial genetic algorithm for multi-objective forest planning. Forest Policy and Economics 21, 12–19.

Gass, S. I., 1987. The setting of weights in linear goal-programming problems. Computers & operations research 14 (3), 227–229.

Greenberg, H., 1971. Integer programming. Vol. 76. Academic Press New York.

Greer, K., Meneghin, B., 1991. Spectrum: An analytical tool for building natural resource management models. US Department of Agriculture Forest Service General Technical Report, 174–178.

Gurobi Optimization, I., 2014. Gurobi optimizer reference manual. Available at: http://www.gurobi.com.

GUSEK, 2013. GLPK Under Scite Extended Kit version 0.2.14. Luiz Bettoni. Boston, USA.

Henderson, D., Jacobson, S. H., Johnson, A. W., 2003. The theory and practice of simulated annealing. In: Handbook of metaheuristics. Springer, pp. 287–319.

Hoganson, H. M., Borges, J. G., 1998. Using dynamic programming and overlapping subproblems to address adjacency in large harvest scheduling problems. Forest Sci. 44 (4), 526–538.

Holland, J. H., 1975. Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence. University of Michigan Press.

Hotvedt, J. E., 1983. Application of linear goal programming to forest harvest scheduling. Southern Journal of Agricultural Economics 15 (1), 103–108.

Johnson, K., Jones, D., 1979. A users guide to multiple use sustained yield resource scheduling calculation (MUSYC). Washington, DC: USDA. Forest Service, Timber Management Staff. 242p.

Johnson, K. N., Stuart, T. W., Crim, S. A., 1986. Forplan version 2: An overview. Washington: USDA Forest Service, 80.

Kaiser, H. M., Messer, K. D., et al., 2011. Mathematical programming for agricultural, environmental and resource economics. John Wiley and Sons, Inc.

Kangas, A., Kangas, J., Kurttila, M., 2008. Decision support for forest management. Vol. 16. Joensuu, Finland. Springer.

Kundur, V., 2005. Prim's algortihm [Matlab Code] Available at: http://www.math.washington.edu/greenbau/math_381/programs/prim.html [Accessed May 10, 2013].

Leuschner, W. A., et al., 1990. Forest regulation, harvest scheduling, and planning techniques. John Wiley & Sons, Inc.

Lockwood, C., Moore, T., 1993. Harvest scheduling with spatial constraints: a simulated annealing approach. Can. J. Forest Res. 23 (3), 468–478.

Lu, F., Eriksson, L. O., 2000. Formation of harvest units with genetic algorithms. Forest Ecol. Man. 130 (1), 57–67.

Maene, C., 2011. Adjacent & neighboring polygons ARCGIS tool box. Available at: http://resources.arcgis. com/gallery/file/geoprocessing [Accessed February 10, 2013].

Mangano, S., 1995. Genetic algorithms solve seemingly intractable problems. Computer Design 34 (5), 70–91.

MATLAB, 2010. version 7.10.0 (R2010a). The MathWorks Inc., Natick, Massachusetts.

Mitchell, M., 1999. An introduction to genetic algorithms. Cambridge, Massachusetts London, England, Fifth printing 3.

Mullen, D. S., 2000. An optimal approach and algorithms for limiting forest opening size in spatially constrained timber harvest scheduling models. USDA. Forest Serv. Gen. Tech. Rep. Nc. 1, 74–80.

Mullen, D. S., Butler, R. M., 2000. The design of a genetic algorithm based spatially constrained timber harvest scheduling model. United States Department of Agriculture Forest Service General Technical Report NC 1, 57–65.

Murray, A. T., Church, R. L., 1995. Heuristic solution approaches to operational forest planning problems. Oper. Res. Spectrum 17 (2), 193–203.

Navon, D. I., 1971. Timber RAM A long-range planning method for commercial timber lands under multiple-use management. U.S. Pacific Southwest Forest and Range Experiment Station. USDA. Forest Service research paper PSW-70 Berkeley, Calif.

Nelson, J., Brodie, J. D., 1990. Comparison of a random search algorithm and mixed integer programming for solving area-based forest plans. Can. J. Forest Res. 20 (7), 934–942.

O'Hara, A. J., Faaland, B. H., Bare, B. B., 1989. Spatially constrained timber harvest scheduling. Can. J. Forest Res. 19 (6), 715–724.

Payne, J. W., 1976. Heuristic search processes in decision making. Advances in consumer research 3 (1), 321–327.

Prim, R. C., 1957. Shortest connection networks and some generalizations. Bell System Technical Journal 36 (6), 1389–1401.

Pukkala, T., Kurttila, M., 2005. Examining the performance of six heuristic optimisation techniques in different forest planning problems. Silva Fenn. 39 (1), 67–80.

Remsoft Inc, 2002. Spatial Woodstock users guide. Frederiction, Canada.

Richards, E. W., Gunn, E. A., 2003. Tabu search design for difficult forest management optimization problems. Can. J. Forest Res. 33 (6), 1126–1133.

Ronnqvist, M., 2003. Optimization in forestry. Math. Program. 97 (1-2), 267–284.

Sassaman, R. W., Holt, E., Bergsvik, K., 1972. User's manual for a computer program for simulating intensively managed allowable cut.

Seymour, R., Hunter, M. L., 1992. New forestry in eastern spruce-fir forests: principles and applications to Maine.

Shan, Y., Bettinger, P., Cieszewski, C., Li, R. T., 2009. Trends in spatial forest planning. Math. Comput. For. Nat.-Res. Sci. 1 (2), Pages–86.

Sherali, H., Soyster, A., 1983. Preemptive and nonpreemptive multi-objective programming: Relationship and counterexamples. Journal of Optimization Theory and Applications 39 (2), 173–186.

Shiffman, D., Fry, S., Marsh, Z., 2012. The Nature of Code. Theoklesia, LLC.

Smaltschinski, T., Seeling, U., Becker, G., 2012. Clustering forest harvest stands on spatial networks for optimised harvest scheduling. Annals of Forest Science 69 (5), 651–657.

Spatial Planning Systems, 2009. What is Patchworks? Available at: http://www.spatial.ca/products/index.html. [Accessed 2014-06-27].

Steuer, R., Schuler, A., 1978. An interactive multiple-objective linear programming approach to a problem in forest management. Operations Research 26 (2), 254–269.

Troncoso, J., D'Amours, S., Flisberg, P., Ronnqvist, M., Weintraub, A., 2011. A Mixed Integer Programming Model to Evaluate Integrating Strategies in the Forest Value Chain: A Case Study in the Chilean Forest Industry. Vol. 28. CIRRELT.

UNCED, 1992. Agenda 21, The Rio Declaration on Environment and Development, the Statement of Forest Principles, the United Nations framework convention on climate change and the United Nations convention on biological diversity. Available at: http://www.un.org/geninfo/bp/enviro.html [Accessed 2014-04-21].

Von Gadow, K., Pukkala, T., Tomé, M., 2000. Sustainable forest management. Vol. 1. Springer.

Wang, J., Geng, Y., Pan, K., 2012. The study on management and development of the value chain construction of state-owned forest resources operational organization. In: Soft Computing in Information Communication Technology. Springer, pp. 531–536.

Weintraub, A., Magendzo, A., Magendzo, A., Malchuk, D., Jones, G., Meacham, M., 1995. Heuristic procedures for solving mixed-integer harvest scheduling-transportation planning models. Can. J. Forest Res. 25 (10), 1618–1626.

Yan, H., 1996. Solving some difficult mixed integer programming problems in production and forest management.

APPENDICES

Appendix A

Chapter 2 Modeling System in GUSEK LP Code

LP Code

This code is also available at http://hdl.handle.net/10402/era.38715

```
set S;
/* stands */

set P;
/* periods */

set nextP within P;
/* periods from period 2 to end */

param a{s in S};
/* area of stands (ha) */

param totarea := sum{s in S} a[s] ;
/* total area of the forest (ha) */

param CVOL{s in S, p in P};
/* conifer yield of stands in period p (m3/ha) */
```

```
param DVOL{s in S, p in P};
/* deciduous yield of stands in period p (m3/ha) */

set M;
/*node _mill */

param r;
/* discount rate */

param midpoint{p in P};
/* years to midpoint of each period */

param harvestcost;
/*harvesting cost  ($/ha)*/

param dist{s in S, m in M};
/* distance between stands and mills */

param price1;
/* price of conifer wood ($/m3) */

param price2;
/* price of deciduous wood ($/m3) */

param dhrevc{s in S, p in P} := CVOL[s,p] * a[s] * price1/ ((1 + r) ^ ...
   ... midpoint[p]);
/* discounted harvest revenue for conifer */

param dhrevd{s in S, p in P} := DVOL[s,p] * a[s] * price2 / ((1 + r) ^ ...
   ... midpoint[p]);
/* discounted harvest revenue for decidious */

param disharvcost{s in S, p in P} := (a[s] * harvestcost)  / ((1 + r) ^ midpoint[p]);
/*discounted harvest cost*/

param dishaulcost{s in S, p in P, m in M}:= ((CVOL[s,p] * a[s] * dist[s,m] * 0.0273)...
   ... +(DVOL[s,p] * a[s] * dist[s,m] * 0.0273)) / ((1 + r) ^ midpoint[p]);
/*discounted haul cost*/
```

```
param adjacent{m in S, n in S} default 0;
/*adjacency table*/

param objcoef{s in S, p in P} := dhrevc[s,p] + dhrevd[s,p] - disharvcost[s,p]...
   ... - sum{m in M} dishaulcost[s,p,m];

/*** DECISION VARIABLES ***/
var x{s in S, p in P} >=0 <=1;

/*** ACCOUNTING VARIABLES ***/
var h_volc{p in P} >= 0;
/* harvest volume in period p (m3)  conifer*/

var h_vold{p in P} >= 0;
/* harvest volume in period p (m3)  deciduous*/

/*** RESOURCE CONSTRAINTS ***/
subject to area{s in S}: sum{p in P} x[s,p] <= 1 ;

/*** ACCOUNTING CONSTRAINTS ***/
s.t. harvvolc{p in P}: sum{s in S} ((CVOL[s,p] * x[s,p] * a[s])) - h_volc[p] = 0;
/* harvest volume (m3) by period */

s.t. harvvold{p in P}: sum{s in S} ((DVOL[s,p] * x[s,p] * a[s])) - h_vold[p] = 0;
/* harvest volume (m3) by period */

/*** POLICY CONSTRAINTS ***/
s.t. vflowc{p in nextP}: h_volc[p] - h_volc[p-1] = 0;
s.t. vflowd{p in nextP}: h_vold[p] - h_vold[p-1] = 0;
/* even-flow constraints */

/*** OBJECTIVE FUNCTION ***/
maximize npv: sum{s in S, p in P} objcoef[s,p] * x[s,p];

solve;
end;
```

```
data;

set P := 1 2 3 4;
set nextP := 2 3 4;

set S := S711341148 S711340912 S711340874 S711346057 S711340927 S711340945...
S711340884 S711340903 S711340958 S711346046 S711340875 S711340899 S711340882...
S711340951 S711340917 S711340946 S711340937 S711340904 S711340869 S711340906...
S711340860 S711346062 S711340895 S711340878 S711340925 S711340885 S711340867...
...<% (37 lines are removed)

set M := M1 M2;

param price1 := 100;
param price2 := 100;

param harvestcost := 3000;

param r := 0.05;

param midpoint :=
1 2.5
2 7.5
3 12.5
4 17.5 ;

param a :=
S711341148 38.1340715
S711340912 39.0473418
S711340874 29.3435650
S711346057 30.6516369
S711340927 10.4496950
S711340945 6.1446621
S711340884 3.3264650
S711340903 2.4952932
S711340958 20.4434707
S711346046 28.2036150
S711340875 10.2478606
... >% (275 lines are removed)
```

```
param CVOL :1 2 3 4 :=
S711341148 32 32 32 32
S711340912 46 46 46 46
S711340874 53 53 53 53
S711346057 0 0 0 0
S711340927 27 27 27 27
S711340945 0 0 0 0
S711340884 39 39 39 39
S711340903 100 100 100 100
S711340958 122 122 122 122
S711346046 0 0 0 0
S711340875 43 43 43 43
S711340899 0 0 0 0
...>% (275 lines  are removed)

param DVOL :1 2 3 4 :=
S711341148 132 132 132 132
S711340912 142 142 142 142
S711340874 158 158 158 158
S711346057 0 0 0 0
S711340927 119 119 119 119
S711340945 0 0 0 0
S711340884 119 119 119 119
S711340903 7 7 7 7
S711340958 10 10 10 10
S711346046 0 0 0 0
S711340875 160 160 160 160
S711340899 0 0 0 0
... >% (275 lines are removed)

param dist :M1 M2 :=
S711341148 5.30669287 1.58678816
S711340912 4.25242897 2.87747645
S711340874 5.24151773 1.21495437
S711346057 5.52641308 0.87711101
S711340927 4.18190148 3.24686740
S711340945 4.02527674 3.73950650
S711340884 4.86040074 1.79522521
```

```
S711340903 4.63372057 2.26455933
S711340958 3.80684916 4.13938629
S711346046 3.73685575 3.28876727
S711340875 4.80245846 1.69942235
S711340899 4.60372162 2.18523906
... >% (275 lines are removed)

param adjacent
S711341148 S711340874 1 ,
S711341148 S711340884 1 ,
S711341148 S711340878 1 ,
S711340912 S711340927 1 ,
S711340912 S711346046 1 ,
S711340912 S711340917 1 ,
S711340912 S711340904 1 ,
S711340912 S711340913 1 ,
S711340912 S711346045 1 ,
S711340874 S711341148 1 ,
S711340874 S711346057 1 ,
... >% (1725 lines are removed)

end;
```

Appendix B

Chapter 2 Modeling System in GUSEK MIP Code

MIP Code

This code is also available at http://hdl.handle.net/10402/era.38715

```
set S;
/* stands */

set P;
/* periods */

set nextP within P;
/* periods from period 2 to end */

param a{s in S};
/* area of stands (ha) */

param totarea := sum{s in S} a[s] ;
/* total area of the forest (ha) */

param CVOL{s in S, p in P};
/* conifer yield of stands in period p (m3/ha) */
```

```
param DVOL{s in S, p in P};
/* deciduous yield of stands in period p (m3/ha) */

set M;
/*node _mill*/

param r;
/* discount rate */

param midpoint{p in P};
/* years to midpoint of each period */

param harvestcost;
/*harvesting cost  ($/ha)*/

param dist{s in S, m in M};
/* distance between stands and mills */

param price1;
/* price of conifer wood ($/m3) */
param price2;
/* price of deciduous wood ($/m3) */

param dhrevc{s in S, p in P} := CVOL[s,p] * a[s] * price1/ ((1 + r) ^ ...
   ... midpoint[p]);
/* discounted harvest revenue for conifer */

param dhrevd{s in S, p in P} := DVOL[s,p] * a[s] * price2 / ((1 + r) ^ ...
   ... midpoint[p]);
/* discounted harvest revenue for decidious */

param disharvcost{s in S, p in P} := (a[s] * harvestcost)  / ((1 + r) ^ midpoint[p]);
/*discounted harvest cost*/

param dishaulcost{s in S, p in P, m in M}:= ((CVOL[s,p] * a[s] * dist[s,m] * 0.0273)...
   ... +(DVOL[s,p] * a[s] * dist[s,m] * 0.0273)) / ((1 + r) ^ midpoint[p]);
/*discounted haul cost*/
```

```
param adjacent{m in S, n in S} default 0;
/*adjacency table*/

param objcoef{s in S, p in P} := dhrevc[s,p] + dhrevd[s,p] - disharvcost[s,p]...
   ... - sum{m in M} dishaulcost[s,p,m];
/* the coefficient of objective function */

/*** DECISION VARIABLES ***/
var x{s in S, p in P} binary;
/* proportion area of stand s cut in period p (ha) */

/*** ACCOUNTING VARIABLES ***/
var h_volc{p in P} >= 0;
/* harvest volume in period p (m3)  conifer*/

var h_vold{p in P} >= 0;
/* harvest volume in period p (m3)  deciduous*/

/*** RESOURCE CONSTRAINTS ***/
subject to area{s in S}: sum{p in P} x[s,p] <= 1 ;

/*** ACCOUNTING CONSTRAINTS ***/
s.t. harvvolc{p in P}: sum{s in S} ((CVOL[s,p] * x[s,p] * a[s])) - h_volc[p] = 0 ;
/* harvest weight (m3) by period */

s.t. harvvold{p in P}: sum{s in S} ((DVOL[s,p] * x[s,p] * a[s])) - h_vold[p] = 0 ;
/* harvest weight (m3) by period */

/*** POLICY CONSTRAINTS ***/
s.t. vflowc{p in nextP}: (h_volc[p]) - 0.95 * (h_volc[p-1]) >= 0;
s.t. evflowc{p in nextP}: (h_volc[p]) - 1.05 * (h_volc[p-1]) <= 0;
/* sequential-flow constraints for conifer harvest volume */

s.t. vflowd{p in nextP}: (h_vold[p]) - 0.95 * (h_vold[p-1]) >= 0;
s.t. evflowd{p in nextP}: (h_vold[p]) - 1.05 * (h_vold[p-1]) <= 0;
/*sequential-flow constraints for deciduous harvest volume */

s.t. adjacency{i in S, p in P}: sum {k in S} adjacent[i,k] * x[k, p] <= 1;
/* adjacency constraints */
```

```
/*** OBJECTIVE FUNCTION ***/
maximize npv: sum{s in S, p in P} objcoef[s,p] * x[s,p];

solve
end;
```

Appendix C

Chapter 2 Modeling System in Gurobi GP Code

Goal Programming Code

```
This code is also available at http://hdl.handle.net/10402/era.38715

set S;
/* stands */

set P;
/* periods */

set nextP within P;
/* periods from period 2 to end */

param a{s in S};
/* area of stands (ha) */

param totarea := sum{s in S} a[s] ;
/* total area of the forest (ha) */

param CVOL{s in S, p in P};
/* conifer yield of stands in period p (m3/ha) */
```

```
param DVOL{s in S, p in P};
/* deciduous yield of stands in period p (m3/ha) */

set M;
/*node _mill_ */

param r;
/* discount rate */

param midpoint{p in P};
/* years to midpoint of each period */

param harvestcost;
/*harvesting cost  ($/ha)*/

param dist{s in S, m in M};
/* distance between stands and mills (m) */

param price1;
/* price of conifer wood ($/m3) */

param price2;
/* price of deciduous wood ($/m3) */

param adjacent{m in S, n in S} default 0;
/*adjacency table*/

param npvgoal;
param w_npv_under;
param w_npv_over;

param charvgoal{p in P};
param w_charv_under{p in P};
param w_charv_over{p in P};

param dharvgoal{p in P};
param w_dharv_under{p in P};
param w_dharv_over{p in P};
param dhrevc{s in S, p in P} := price1 * CVOL[s,p] * a[s] / ((1 + r) ^ ...
   ... midpoint[p]);
/* discounted harvest revenue for conifer */

param dhrevd{s in S, p in P} := price2 * DVOL[s,p] * a[s] / ((1 + r) ^ ...
   ... midpoint[p]);
/* discounted harvest revenue for decidious */

param disharvcost{s in S, p in P} := (a[s] * harvestcost)  / ((1 + r) ^ midpoint[p]);
/*discounted harvest cost*/

param dishaulcost{s in S, p in P, m in M} := ((CVOL[s,p] * a[s] *dist[s,m] * 0.0273)...
   ... +(DVOL[s,p] * a[s] * dist[s,m] * 0.0273)) / ((1 + r) ^ midpoint[p]);
/*discounted haul cost*/

param objcoef{s in S, p in P} := dhrevc[s,p] + dhrevd[s,p] - disharvcost[s,p]...
   ... - displantcost[s,p] - sum{m in M} dishaulcost[s,p,m];

/*** DECISION VARIABLES ***/
var x{s in S, p in P} binary;
/* proportion area of  stand s cut in period p (ha) */

/*** GOAL VARIABLES ***/
var npvunder >= 0;
var npvover >= 0;
var charvunder {p in P} >= 0;
var charvover {p in P} >= 0;
var dharvunder {p in P} >= 0;
var dharvover {p in P} >= 0;

/*** ACCOUNTING VARIABLES ***/
var h_volc{p in P} >= 0;
/* harvest volume in period p (m3) */

var h_vold{p in P} >= 0;
/* harvest volume in period p (m3) */

var npv;
/* net present value */
```

```
/*** RESOURCE CONSTRAINTS ***/
subject to area{s in S}: sum{p in P} x[s,p] <= 1 ;

/*** ACCOUNTING CONSTRAINTS ***/
s.t. harvvolc{p in P}: sum{s in S} (CVOL[s,p] * x[s,p] * a[s]) - h_volc[p] = 0 ;
/* conifer harvest weight (m3) by period */

s.t. harvvold{p in P}: sum{s in S} (DVOL[s,p] * x[s,p] * a[s]) - h_vold[p] = 0 ;
/* deciduous harvest weight (m3) by period */

s.t. npvcalc: sum{s in S, p in P} objcoef [s,p] * x[s,p] - npv =0;

/*** POLICY CONSTRAINTS ***/
s.t. adjacency{i in S, p in P}: sum {k in S} adjacent[i,k] * x[k, p] <= 1;

/*** GOAL CONSTRAINTS ***/
s.t. npvgoal_c : npv - npvover + npvunder = npvgoal;

s.t. charvgoal_c {p in P}: h_volc[p] - charvover[p] + charvunder[p] = charvgoal[p];

s.t. dharvgoal_c {p in P}: h_vold[p] - dharvover[p] + dharvunder[p] = dharvgoal[p];

/*** OBJECTIVE FUNCTION ***/
minimize wdevgoal: w_npv_under * npvunder + w_npv_over * npvover + sum{p in P}...
   ... (w_charv_under[p] * charvunder[p] + w_charv_over[p] * charvover[p] + ...
   ...  w_dharv_under[p] * dharvunder[p] + w_dharv_over[p] * dharvover[p])

solve;
end;
data;

param npvgoal := 2e+07;
param w_npv_under := 1;
param w_npv_over := 0;

param charvgoal :=
1 15000
2 15000
```

```
3 15000
4 15000;
param w_charv_under :=
1 1333
2 1333
3 1333
4 1333;
param w_charv_over :=
1 1333
2 1333
3 1333
4 1333;

param dharvgoal :=
1 25000
2 25000
3 25000
4 25000;
param w_dharv_under :=
1 800
2 800
3 800
4 800;
param w_dharv_over :=
1 800
2 800
3 800
4 800;
end;
```

Appendix D

Chapter 3 Modeling System in Matlab GA Code

Genetic Algorithm Code

```
This code is also available at http://hdl.handle.net/10402/era.38715

% a sample code of genetic algorithm
clear all;
tic;

utm=xlsread('latlong.xlsx');
northing=utm(:,3);
easting=utm(:,2);
adjnorthing=northing-6.108e6;
adjeasting=easting-436e3;

load('forest.mat')

ngenerations=500;
nperiods=4;
nbreeding=100;

[nstands,t]=size(area);
standlist  = (1:nstands)';
```

```
totvol=cvol(:,1)+dvol(:,1);
forested=zeros(nstands,1);
t=find(totvol > 0);
forested(t,1)=1;
k=find(totvol <=0);
nonforested(k,1)=1;

dprice=100;
cprice=100;
logcost=2000;
chaulcost=0.0273;
dhaulcost=0.0273;
discrate=0.05;
midpoint=[2.5; 7.5; 12.5; 17.5];

mutationrate=0.005;

initialpop=randi([0,nperiods],nbreeding,nstands);
initialpop(:,k)=0;
cstandvol=cvol(:,1) .* area;
dstandvol=dvol(:,1) .* area;

npvgoal=2e7;
npvpnlto = 0;
npvpnltu = 1;

cvolgoal=[15000 15000 15000 15000];
cvolpnlto = 1e3;
cvolpnltu = 1e3;

dvolgoal=[25000 25000 25000 25000];
dvolpnlto = 1e3;
dvolpnltu = 1e3;

adjgoal = 0;
adjpnlto = 1e6;
adjpnltu = 0;
```

```
resmstgoal = 0;
resmstpnlto = 1e3;
resmstpnltu = 0;

resvaluegoal=200;
resvaluepnlto=0;
resvaluepnltu=1e5;

oldpop = initialpop;
penaltyarray=[];
for i = 1:nbreeding
    sched=oldpop(i,:);

    calcpenalty;
    penaltyarray=[penaltyarray penalty];
    %{i, totalcon, totaldec, npv, adjacencyviolations,reservearea, mstcost, penalty}

end

penaltylist=[];

maxpenalty=max(penaltyarray);
minpenalty=min(penaltyarray);
fitness=(maxpenalty - penaltyarray)/(maxpenalty-minpenalty);

bestsched=[];
bestpenalties=[];
bestnpv=[];
bestcvol=[];
bestdvol=[];
bestresarea=[];
bestresmst=[];
bestrestri=[];
bestadj=[];
bestresvalue=[];

for igen = 1:ngenerations
    {igen,toc}
```

```
breeders=[];
newpop=[];
penaltyarray=[];
npvarray=[];
resareaarray=[];
resmstarray=[];
adjarray=[];
cvolarray=[];
dvolarray=[];
triarray=[];
resvaluearray=[];

for i=1:nbreeding
    findapair;
    breeders=[breeders;parents];
    crossover;
    mutate1=rand(1,nstands); % random number for comparison with mutation rate
    mutate2=mutate1 <= mutationrate; % identifies stands which mutate
    % if the stand mutates, this is what it would mutate to.
    mutate3=randi([0,nperiods],1,nstands);
    mutate4=child1 - mutate2 .* child1; % zeros the stands that mutate
    child1=mutate4 + mutate2 .* mutate3; % puts the mutation in
    child1(:,k)=0;

    mutate1=rand(1,nstands);
    mutate2=mutate1 <= mutationrate;
    mutate3=randi([0,nperiods],1,nstands);
    mutate4= child2 - mutate2 .* child2;
    child2=mutate4 + mutate2 .* mutate3;
    child2(:,k)=0;

    sched=child1;
    calcpenalty;
    child1penalty=penalty;
    child1npv=npv;
    child1resarea=reservearea;
    child1mstcost=mstcost;
    child1adj=adjacencyviolations;
    child1conharvest=conharvest';
```

```
child1decharvest=decharvest';
child1tri=tri;
child1resvalue=reservevalue;

sched=child2;
calcpenalty;
child2penalty=penalty;
child2npv=npv;
child2resarea=reservearea;
child2mstcost=mstcost;
child2adj=adjacencyviolations;
child2conharvest=conharvest';
child2decharvest=decharvest';
child2tri=tri;
child2resvalue=reservevalue;

if child1penalty <= child2penalty
    newpop=[newpop;child1];
    penaltyarray=[penaltyarray child1penalty];
    npvarray=[npvarray child1npv];
    resareaarray=[resareaarray child1resarea];
    resmstarray=[resmstarray child1mstcost];
    adjarray=[adjarray child1adj];
    cvolarray=[cvolarray;child1conharvest];
    dvolarray=[dvolarray;child1decharvest];
    tevpop(j).penalty=child1penalty;
    resvaluearray=[resvaluearray child1resvalue];

else
    newpop=[newpop;child2];
    penaltyarray=[penaltyarray child2penalty];
    npvarray=[npvarray child2npv];
    resareaarray=[resareaarray child2resarea];
    resmstarray=[resmstarray child2mstcost];
    adjarray=[adjarray child2adj];
    cvolarray=[cvolarray;child2conharvest];
    dvolarray=[dvolarray;child2decharvest];
    tevpop(j).penalty=child2penalty;
    resvaluearray=[resvaluearray child2resvalue];
```

```
        end
    end

    penaltylist=[penaltylist; penaltyarray];

    maxpenalty=max(penaltyarray);
    minpenalty=min(penaltyarray);
    fitness=(maxpenalty - penaltyarray)/(maxpenalty-minpenalty);

    %    newpenalties
    [minpenalty,ndx]=min(penaltyarray);
    bestpenalties=[bestpenalties;minpenalty];
    bestsched=[bestsched;newpop(ndx,:)];
    bestnpv=[bestnpv;npvarray(ndx)];
    bestresarea=[bestresarea;resareaarray(ndx)];
    bestresmst=[bestresmst;resmstarray(ndx)];
    bestadj=[bestadj;adjarray(ndx)];
    bestcvol=[bestcvol;cvolarray(ndx,:)];
    bestdvol=[bestdvol;dvolarray(ndx,:)];
    bestresvalue=[bestresvalue; resvaluearray(ndx)];
    %    breeders
    oldpop=newpop;
end

[minminpenalty,d]=min(bestpenalties); %lowest penalty in the bestpenalty array
son.sched=bestsched(d,:),
son.sched
son.penalty=minminpenalty;
son.npv=bestnpv(d,:);
son.resarea=bestresarea(d,:);
son.resmst=bestresmst(d,:);
son.adj=bestadj(d,:);
son.cvol=bestcvol(d,:);
son.dvol=bestdvol(d,:);
son.resvalue=bestresvalue(d,:);

son
```

```
bestschedule=bestsched(d,:);

todd=min(penaltylist’);

figure;
plot(bestpenalties’) %figure2 of minimum penalties of each generation
xlabel(’Generation’);
ylabel(’Minimum Penalties’);
set(gcf,’numbertitle’,’off’,’name’,’minimum penalties’);

figure
plot(bestnpv’)
xlabel(’Generation’)
ylabel(’Net Present Value’)
set(gcf,’numbertitle’,’off’,’name’,’NPV’)

figure
plot(sum(bestcvol’))
xlabel(’Generation’)
ylabel(’Harvest Volume’)
set(gcf,’numbertitle’,’off’,’name’,’harvest volume of conifer’)

figure
plot(sum(bestdvol’))
xlabel(’Generation’)
ylabel(’Harvest Volume’)
set(gcf,’numbertitle’,’off’,’name’,’harvest volume of deciduous’)

figure
plot(bestadj)
xlabel(’Generation’)
ylabel(’Adjacency Violation’)
set(gcf,’numbertitle’,’off’,’name’,’Adjacency violation2’)

figure
plot(bestresvalue)
xlabel(’Generation’)
ylabel(’Reserve Value’)
set(gcf,’numbertitle’,’off’,’name’,’Reserve Value’)
```

```
figure
plot(bestresmst)
xlabel('Generation')
ylabel('Minimum Spanning Tree (m)')
set(gcf,'numbertitle','off','name','MST')

dlmwrite('harvsched.csv', bestschedule', 'newline', 'pc')
dlmwrite('penaltylist.csv', penaltylist, 'newline','pc')
dlmwrite('standardsapma.csv', aaa', 'newline','pc')
dlmwrite('reservevalue2.csv', bestresvalue, 'newline','pc')
dlmwrite('bestpenaltylist.csv', bestpenalties, 'newline','pc')

toc;

---------------------------------------
%  Penalty Function  %

harvsched=zeros(nperiods,nstands);
for j=1:nperiods
    t=find(sched == j);
    harvsched(j,t)=1;
end
conharvest=harvsched * cstandvol;
totalcon=sum(conharvest);
cvolo=max(0,conharvest'-cvolgoal);
cvolu=max(0,cvolgoal-conharvest');

decharvest=harvsched * dstandvol;
totaldec=sum(decharvest);
dvolo=max(0,decharvest'-dvolgoal);
dvolu=max(0,dvolgoal-decharvest');

adjacencyviolations = ...
    sum(diag(harvsched * tril(adjacencytable) * harvsched'));
adjo=max(0,adjacencyviolations-adjgoal);
adju=max(0,adjgoal-adjacencyviolations);

harvrev=harvsched *(cprice * cstandvol + dprice * dstandvol);
```

```
harvcost=harvsched * (logcost * area);
haulcost=harvsched*((distancesawmill .* (chaulcost * cstandvol)) + ...
   ... (distancepulpmill .* (dhaulcost * dstandvol)));

npv = sum((harvrev - harvcost - haulcost) ./ (1 + discrate) .^ midpoint);
npvo = max(0,npv - npvgoal);
npvu = max(0,npvgoal - npv);

t=find(sched==0);
unharvested=zeros(nstands,1);
unharvested(t,1)=1;
reserve= unharvested .* forested;

reservearea=sum(reserve .* area);

if sum(reserve) <=2
    mstcost=0;
else
[mstcost,tri] = calcmst( standlist, reserve,adjnorthing,adjeasting);
resmsto = max(0,mstcost-resmstgoal);
resmstu = max(0,resmstgoal-mstcost);
end

reservevalue= sum(reservequality .* (reserve.*area));
resvalueo=max(0,reservevalue-resvaluegoal);
resvalueu=max(0,resvaluegoal-reservevalue);

penalty = ...
    npvpnlto * npvo + npvpnltu * npvu + ...
    sum(cvolpnlto * cvolo) + sum(cvolpnltu * cvolu) + ...
    sum(dvolpnlto * dvolo) + sum(dvolpnltu * dvolu) + ...
    adjpnlto * adjo + adjpnltu * adju + ...
    resmstpnlto * resmsto + resmstpnltu * resmstu + ...
    resvaluepnlto * resvalueo + resvaluepnltu * resvalueu;
---------------------------------------
```

```
%  Function to Find Pair  %

breed=0;
cntr=0;
while breed < 2
    cntr=cntr+1;
    parents=randi([1,nbreeding],1,2);
    fit=fitness(parents);
    chance=rand(1,2);
    breed=sum(chance < fit);
end;
parent1=oldpop(parents(1,1),:);
parent2=oldpop(parents(1,2),:);

---------------------------------------
%  Crossing over function  %

xover=randi([1,nstands-1],1);
child1=[parent1(1:xover) parent2(xover+1:nstands)];
child2=[parent2(1:xover) parent1(xover+1:nstands)];

---------------------------------------
%  Minimum Spanning Tree Function  %

function [cost, tri] = calcmst( standlist, reserve,adjnorthing,adjeasting)
if sum(reserve) <= 2
    cost=0;
    mst=[];
else
    reservestandlist = standlist .* reserve;
    newreservestandlist=reservestandlist(reservestandlist ~= 0);
    north2 = adjnorthing(newreservestandlist,:);
    east2 = adjeasting(newreservestandlist,:);
    tri=DelaunayTri(horzcat(east2,north2));
    edges=unique(sort(...
        [tri(:,2) tri(:,1);tri(:,3) tri(:,2);tri(:,1) tri(:,3)],2),'rows');
    point1=edges(:,1);
    point2=edges(:,2);
    p1=tri.X(point1,:);
    p2=tri.X(point2,:);
    edgedistance=sqrt((p1(:,1) - p2(:,1)).^2 + (p1(:,2) - p2(:,2)).^2);
    t=size(edges);
    nedges=t(1,1);
    t=size(tri.X);
    nvertices=t(1,1);
    newadjacencymatrix=ones(nvertices)*1e6;
    for i = 1:nedges
        newadjacencymatrix(edges(i,2),edges(i,1))=edgedistance(i);
        newadjacencymatrix(edges(i,1),edges(i,2))=edgedistance(i);
    end
    [mst,cost]=prim(newadjacencymatrix);
end
end
---------------------------------------
%  Prim's Algorithm for Minimum Spaning Tree Function  %

function [mst, cost] = prim(A)

% User supplies adjacency matrix A.  This program uses Prim's algorithm
% to find a minimum spanning tree.  The edges of the minimum spanning
% tree are returned in array mst (of size n-1 by 2), and the total cost
% is returned in variable cost.  The program prints out intermediate
% results and pauses so that user can see what is happening.  To continue
% after a pause, hit any key.

[n,n] = size(A); % The matrix is n by n, where n = # nodes.

if norm(A-A','fro') ~= 0 ,% If adjacency matrix is not symmetric,
  disp('Error:Adjacency matrix must be symmetric')% print error message and quit.
  return,
end;

% Start with node 1 and keep track of which nodes are in tree and which are not.

intree = [1];  number_in_tree = 1;  number_of_edges = 0;
notintree = [2:n]';  number_notin_tree = n-1;

in = intree(1:number_in_tree); % Print which nodes are in tree and which
```

```
out = notintree(1:number_notin_tree);% pause, % are not.

% Iterate until all n nodes are in tree.

while number_in_tree < n,
%   Find the cheapest edge from a node that is in tree to one that is not.
  mincost = Inf;% You can actually enter infinity into Matlab.
  for i=1:number_in_tree,
    for j=1:number_notin_tree,
      ii = intree(i);  jj = notintree(j);
      if A(ii,jj) < mincost,
        mincost = A(ii,jj); jsave=j; iisave=ii; jjsave=jj;% Save coords of node.
      end;
    end;
  end;

%    Add this edge and associated node jjsave to tree. Delete node jsave from list
%    of those not in tree.

  number_of_edges = number_of_edges + 1;% Increment number of edges in tree.
  mst(number_of_edges,1) = iisave;% Add this edge to tree.
  mst(number_of_edges,2) = jjsave;
  costs(number_of_edges,1) = mincost;

  % Increment number of nodes that tree connects.
  number_in_tree = number_in_tree + 1;
  intree = [intree; jjsave];% Add this node to tree.
  for j=jsave+1:number_notin_tree,%Delete this node from list of those not in tree.
    notintree(j-1) = notintree(j);
  end;
  % Decrement number of nodes not in tree.
  number_notin_tree = number_notin_tree - 1;

  in = intree(1:number_in_tree);% Print which nodes are now in tree and
  out = notintree(1:number_notin_tree); %pause,% which are not.

end;
% Print out edges in minimum spanning tree.
cost = sum(costs);
```

```
---------------------------------------
%  Area  %

38.13407150440
39.04734178520
29.34356498380
30.65163688380
10.44969497560
6.14466210933
3.32646504784
2.49529321907
20.44347068040
28.20361504650
... >% (277 lines are removed)
---------------------------------------
%  Conifer Harvest Volume (m3/ha)  %

32 32 32 32
46 46 46 46
53 53 53 53
0 0 0 0
27 27 27 27
0 0 0 0
39 39 39 39
100 100 100 100
122 122 122 122
0 0 0 0
... >% (277 lines are removed)
---------------------------------------
%  Deciduous Harvest Volume (m3/ha)  %

132 132 132 132
142 142 142 142
158 158 158 158
0 0 0 0
119 119 119 119
0 0 0 0
119 119 119 119
7 7 7 7
```

```
10 10 10 10
0 0 0 0
... >% (277 lines are removed)
---------------------------------------
%  Distance between pulp mill and stands (km)  %

1.58678816
2.87747645
1.21495437
0.87711101
3.2468674
3.7395065
1.79522521
2.26455933
4.13938629
3.28876727
... >% (277 lines are removed)
---------------------------------------
%  Distance between sawmill and stands (km)  %

5.30669287
4.25242897
5.24151773
5.52641308
4.18190148
4.02527674
4.86040074
4.63372057
3.80684916
3.73685575
... >% (277 lines are removed)
---------------------------------------
%  Adjacency Matrix (stand by stand)  %

0,0,1,0,0,0,1,0,0,0,
0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,
... >% (277 x 277 lines are removed)
---------------------------------------
% Reserve quality weighted area  (ha) %

0.375
0.25
0.625
0
0.25
0
0.125
0.25
0.25
0
... >% (277 lines are removed)
---------------------------------------
%  Coordinatres of each stand %

toy_ID POINT_X POINT_Y
711341148  439472.36618000000 6112703.62400000000
711340912   438416.26017400000 6111910.27951000000
711340874  439735.06329900000 6112258.07666000000
711346057  440803.62469700000 6112106.66794000000
711340927   437874.77106600000 6112179.47122000000
711340945  437347.68906200000 6111997.55570000000
711340884   439428.19559500000 6112113.57700000000
711340903   438861.21288700000 6112274.12937000000
711340958   436974.40555000000 6111893.73565000000
... >% (277 lines are removed)
---------------------------------------
```