ISSN 1946-7664. MCFNS 2012
Vol. 4, Issue 1, pp. 16–26
Mathematical and Computational Forestry & Natural-Resource Sciences

AVAILABLE ONLINE AT HTTP://MCFNS.COM

Submitted: Oct. 11, 2010
Accepted: Aug. 23, 2011
Published: Feb. 28, 2012
Last Correction: Aug. 23, 2011

OPTIMAL PARAMETER SETTINGS FOR SOLVING HARVEST SCHEDULING MODELS WITH ADJACENCY CONSTRAINTS

Phillip J Manning[1], Marc E McDill[2]

[1] *Northeast Land Management, LLC, Harrisburg, PA 17111 USA*

[2] *College of Agricultural Sciences, Penn State, University Park, PA 16802 USA*

Abstract. Optimal parameter settings to improve the efficiency of solving harvest scheduling models with adjacency constraints were examined using Ilog's Cplex® 11.2 optimizer tuning tool. A total of 160 randomly generated hypothetical forests were created with either 50 or 100 stands and four age-class distributions. Mixed integer programming problems were formulated in Model I form with four different adjacency constraint types, two Unit Restriction Model (URM) adjacency constraints (Pairwise and Maximal Clique) and two Area Restriction Model (ARM) formulations (Path and Generalized Management Unit). A total of 640 problem sets—where a set is a common forest size, age-class distribution, and adjacency constraint type—were tuned to determine optimal parameter settings and then were solved at both the default and optimal settings. In general, mean solution time was less for a given problem set using the optimal parameters compared to the default parameters. The results discussed provide a simple approach to decrease the solution time of solving mixed integer forest planning problems with adjacency constraints.

Keywords: Harvest scheduling, adjacency constraints, forest management, area restriction models, unit restriction models.

1 Introduction

Forest management planning has become increasingly complex over the last two decades due to growing sustainability concerns and non-timber objectives. Optimization techniques have been used successfully to schedule harvests since the 1960's (Thompson et al. 1973). Early models were typically formulated as linear programming problems. Modern management guidelines, however, often require spatial restrictions on the size and proximity of harvest openings, commonly called adjacency constraints, along with other spatial constraints such as those related to wildlife habitat conservation. A common approach to handling adjacency restrictions is to impose constraints that prevent adjacent stands from being harvested within a given time period (McDill and Braze 2000). Such restrictions add complexity to the task of finding optimal solutions because available harvest scheduling formulations that impose them require binary decision variables, making the harvest scheduling problem with spatial constraints an integer or mixed integer programming (MIP) problem (Crowe et al. 2003). These models are substantially more difficult to solve than similar-sized harvest scheduling models formulated as linear programming problems (Williams 1993; McDill and Braze 2001).

Two broad types of adjacency constraints have been discussed for incorporating spatial management requirements in harvest scheduling models, (i) the Unit Restriction Model (URM) and (ii) the Area Restriction Model (ARM) (Murray 1999). The URM predefines the boundaries of all management units and restricts concurrent harvest on all adjacent units regardless of their size (Murray 1999). A number of exact formulation approaches have been proposed to solve this model, including the branch and bound algorithm (BBA; McDill and Braze 2001), column generation (Barahona et al. 1992; Weintraub et al. 1994), and dynamic programming (Hoganson and Borges 1998). In addition, heuristic methods such as Tabu search (Murray and Church 1995; Boston and Bettinger 1999) and simulated annealing (O'Hare et al. 1989; Nelson and Brodie 1990; Lockwood and Moore 1993; Murray and Church 1995; Boston and Bettinger 1999) have been used. In contrast, the ARM does not predefine the boundaries of harvest blocks; instead, small management units may be combined into larger harvest blocks as long as their com-

bined area does not exceed a given maximum (Murray 1999). The ARM is a more complex problem than the URM, but the added complexity can be worthwhile. For instance, Murray and Weintraub (2002) noted that net revenues can be improved through cluster forming, and Richards and Gunn (2000) found that having predefined units as in the URM may underestimate the potential harvest flow across suboptimal units. Much of the literature related to solving the ARM problem has focused on heuristic methods (Lockwood and Moore 1993; Barrett et al. 1998; Richards and Gunn 2000; Boston and Bettinger 2002; Caro et al. 2003) for solving these problems. However, several exact MIP formulations of the ARM problem have been proposed, including the Path formulation (McDill et al. 2002), the Generalized Management Unit (GMU, McDill et al. 2002) or Cluster Packing formulation (Goycoolea et al. 2005), and the Bucket formulation (Constantino et al. 2006). The heuristic methods discussed for both the URM and ARM problems are good for finding near-optimal solutions for large complex problems in a relatively small amount of time. Furthermore, exact formulations for some problems, for example some habitat problems, have not been developed and may be difficult, if not impossible to develop (e.g., Bettinger et al. 1999). However, heuristic approaches typically cannot guarantee that the solution found is optimal or how close it is to being optimal (McDill and Braze 2001). On the other hand, the BBA provides a general method to obtain an exact solution to MIP problems (McDill and Braze 2001). In actuality, the BBA, as implemented in most modern software, such as Ilog's Cplex§, includes heuristic algorithms in the solution process. Hence, the key distinction of the BBA is that it provides guaranteed bounds on the objective function value of the final solution.

McDill and Braze (2001) examined the BBA as a means to solve harvest scheduling models with URM type adjacency constraints. They found that moderately large problems with immature or balanced age-class distributions could be solved to optimality on average in 7.6 to 17.4 min, respectively, while overmature and old-growth forests took in excess of 4 hr to solve to optimality (McDill and Braze 2001). They also discussed the advantages of being able to use off-the-shelf software packages such as Ilog's Cplex§to solve forest planning problems using the BBA, including cost, flexibility, and continued improvement in software development (McDill and Braze 2001). Crowe et al. (2003) extended the work of McDill and Braze (2001) to harvest scheduling models with ARM adjacency constraints. Crowe et al. (2003) found that small to medium sized problems could be solved to optimality or near optimality in a reasonable time using the BBA. Both of these studies, however, solved the harvest scheduling problems using the default parameter settings, specifying only the desired optimal tolerance gap. Little work has been done to identify optimal parameter settings for solving URM and ARM problems with the BBA. Crowe et al. (2003) expressed the need for further research to identify the influence various BBA parameters have on the efficiency of solving ARM problems and, for that matter, all spatially explicit forest planning problems.

This article examines some of the parameters associated with the BBA and evaluates their settings using the performance tuning tool in Cplex§. Our objectives are (i) to provide a review of topics associated with parameters of the performance tuning tool, (ii) to identify parameter settings that will improve solution time for spatially explicit forest planning problems with different sizes, age-class distributions, and adjacency constraints, and (iii) to assess the role of the parameter tuner in solving more complex forest planning problems.

2 Parameter Tuning

Various parameter settings are available when solving MIP problems using the BBA in Cplex§. The performance of the solver can depend significantly on the values of these parameters. The default setting for the majority of these parameters is that they are set automatically by the optimizer based on an analysis of the problem. To be able to understand how changes in parameters affect solving MIP problems in Cplex§, one must have a general understanding of the BBA. Cplex§implements the BBA by solving a series of subproblems that are relaxations of the MIP where some or all of the binary variables are treated as continuous variables bounded between zero and one. Each relaxation is a node of the branch and bound tree (ILOG 2008). The root of the tree is the continuous relaxation of the original MIP problem (ILOG 2008). If the solution to the relaxation at a given node has one or more fractional variables, cuts may be implemented to eliminate areas of the feasible region of the relaxation that contain the fractional solutions without eliminating any feasible integer solutions (ILOG 2008). If fractional variables exist after applying the cuts, the algorithm branches on a remaining fractional integer variable to generate two new subproblems with the branching variable fixed at either zero or one (ILOG 2008). These subproblems may result in an all-integer solution, an infeasible solution, or another fractional solution. If an all-integer solution is found that is superior to the current best all-integer solution, it establishes a new primal bound. If another fractional solution is produced that is superior to the current primal bound, the process is repeated. If a fractional solution is produced that is inferior to the current primal bound, the branch is fathomed from the tree –

i.e., not considered further. McDill and Braze (2001) provide a more detailed explanation of the BBA along with an example of its implementation. The following sections describe parameters that have notable effects on solving MIP problems with Cplex§.

2.1 Preprocessing Preprocessing is completed automatically by the Cplex§presolver and aggregator one or more times to strengthen the initial linear relaxation and to decrease the overall size of the MIP (ILOG 2008). Bound strengthening tightens the bounds on variables, possibly leading to fixing some variables and thus removing them from consideration during the BBA. This procedure can take a long time. However, the reduction is usually beneficial to solving the problem (ILOG 2008). This process can be controlled using the set preprocessing boundstrength parameter. The default setting of this parameter is -1, which means that the preprocessor automatically determines whether or not to apply bound strengthening. Bound strengthening can be turned off with a setting of 0 and applied with a setting of 1. Constraint tightening – i.e., the reduction of constraint coefficients – is handled by the set preprocessing coeffreduce parameter. This parameter determines how coefficient reduction is applied; where a value of 2 (default) reduces all potential coefficients, 1 reduces only integral coefficients, and 0 applies no reductions (ILOG 2008). Coefficient reduction usually strengthens the continuous relaxation and reduces the number of nodes in the BBA tree, although sometimes it increases the amount of time needed to solve the linear relaxation at each node thus offsetting the benefit of fewer nodes. In addition, the set preprocessing relax control determines whether an LP pre-solve is applied to the root relaxation of the MIP problem (ILOG 2008). The LP pre-solve on the relaxation can produce additional reductions in the MIP beyond the other preprocessing functions. Finally, the set preprocessing numpass and set preprocessing aggregator limit the number of analysis passes the presolver and aggregator make, respectively.

2.2 Probing Probing examines the implications of fixing each binary variable to zero or one prior to actually fixing any variables (ILOG 2008). Probing occurs after preprocessing but before the BBA is implemented (ILOG 2008). According to ILOG (2008), the probing feature can help in solving some problems but it is costly due to the time it takes to probe, which may or may not pay off with shorter overall solution times. This parameter can be set with the set mip strategy probe i, where i is the parameter value. The default value of zero automatically determines the level of probing, positive values set higher levels of probing, and a value of -1 turns off probing (ILOG 2008).

2.3 Cuts A cut is a constraint that can be added to the model to eliminate non-integer solutions that would otherwise be feasible for the relaxation at a given node (ILOG 2008). Adding cuts usually reduces the number of branches needed to solve the MIP. Cuts occur more frequently at the root node, but they can be added by the optimizer at other nodes depending on the problem (ILOG 2008). Cplex§generates cuts that are valid for all subproblems (ILOG 2008). The types of cuts available in Cplex §are described in Table 1. Under the default setting (zero), the optimizer automatically determines how often, if at all, to generate each class of cut. A setting of -1 indicates that no cuts of that class should be generated, while

settings of one or two generate cuts moderately and aggressively, respectively (ILOG 2008). Clique, cover, and disjunctive cuts permit a setting of three, which specifies that the class of cut should be generated very aggressively (ILOG 2008).

2.4 Heuristics Determining good, but not necessarily optimal, feasible solutions quickly during the BBA improves the primal bound and may decrease the time required to find and prove that a solution is optimal or to reach a specified optimality gap. Two heuristics can be employed to find integer solutions at nodes during the BBA. The first type is the node heuristic, which tries to construct a feasible solution from the fractional solution at a node and is controlled by the set mip strategy heuristicfreq parameter (ILOG 2008). Its default is to dynamically determine how often to apply the heuristic is (ILOG 2008). Any positive value of the parameter specifies the frequency, in node count, to apply the heuristic (ILOG 2008). For instance, a parameter value of 20 would apply the heuristic at every 20^{th} node during optimization. The second type of heuristic available in Cplex§is the Relaxation Induced Neighborhood Search (RINS). This heuristic explores a neighborhood of the current solution to try to find a new and improved solution by formulating the neighborhood as a subMIP (ILOG 2008). Two parameters are associated with the RINS. The set mip strategy rinsheur controls how often to invoke the RINS heuristic while the submipnodelim parameter restricts the number of nodes to search in the subMIP during the RINS (ILOG 2008).

2.5 Performance tuning tool The parameter tuning tool available in Ilog's Cplex§11.2 optimizer is a utility to aid in improving the performance of solving MIP problems (ILOG 2008). Performance refers to decreasing the time it takes to solve a problem to optimality or a specified optimality gap. Modern MIP solvers provide great flexibility due to the available parameters one can change. However, the number of possible param-

Table 1: Description of the types of cuts available in Cplex§.

Cut type	Interactive optimizer command	Description
Clique	set mip cuts cliques	Relationship between a group of binary variables such that at most one variable can be positive in any integer feasible solution.
Cover	set mip cuts covers	Constraint represents a knapsack constraint then there is a minimal cover associated with it and a constraint can be constructed based on this condition.
Disjunctive	set mip cuts disjunctive	Inequalities that represent valid feasible regions of the LP relaxations of the subproblems but are not valid for the feasible region of the LP relaxation of the MIP problem.
Flow cover	set mip cuts flowcovers	Generated from constraints with continuous variables where the continuous variables have variable upper bounds that are zero or positive depending on the setting of the associated binary variable.
Flow path	set mip cuts path	Generated based on a set of constraints containing the continuous variables that define a path structure in a network where the constraints are nodes and the continuous variables are flows that will be on or off depending on the settings of the binary variables.
Gomory	set mip cuts gomory	Generated based on applying integer rounding on a pivot row in the optimal LP tableau for a basic integer variable with a fractional solution.
GUB cover	set mip cuts gubcovers	A constraint for a set of binary variables whose sum of variables is less than or equal to one. GUB cover cuts are stronger than ordinary cover cuts if the variables in the GUB are also members of a knapsack constraint then the minimal cover can be selected with the additional consideration that at most one member of the GUB can be one in a solution.
Implied bound	set mip cuts implied	Cuts which reflect the relationship between binary variables and their implied bounds on continuous variables.
MIR	set mip cuts mircut	Applied based on integer rounding on the coefficients of integer variables and the right hand side of a constraint.
Zero-half	set mip cuts zerohalfcut	Generated based on the observation that when the left hand side of an inequality consists of integer variables and coefficients then the right hand side can be rounded down

eter combinations is large. For instance, consider five parameters in Cplex§, four parameters with three values each and two parameters with four values each producing 1,296 possible combinations, far too many to manually examine and test every possible combination. The performance tuning tool provides a method to systematically examine the parameters and determine the best settings for the given problem or set of problems. The tool will not, however, correct models that suffer from instability or insufficient memory but will discern performance issues and suggest a suite of parameter settings that lead to faster solution times (ILOG 2008). The tool accomplishes this through a two-phase process. The first phase is to solve the problem, or set of problems, at the default parameter settings. In the second phase, the tuning tool performs several optimizations at various parameter settings to determine which settings are appropriate for the given problem to improve performance. The remaining sections of this article assess the ability of the tuning tool to improve the efficiency of solving spatially explicit forest planning problems.

3 Model Formulation

A total of 160 random hypothetical forests were created with the MakeLand program (McDill and Braze 2000). MakeLand creates a set of random, contiguous polygons (stands) and randomly assigns an age-class to each polygon based on a target age-class distribution (McDill and Braze 2000). For this study, forests were created with 50 (Figure 1) and 100 (Figure 2) polygons to represent small and large forests. Each stand is assigned a stand number and an initial age-class as shown in Figures 1 and 2. The average polygon size was 20 ha. Thus the 50- and 100-stand forests have total areas of 1000 and 2000 ha, respectively. The stands in these forests all have the same forest type and site quality; they differ only in terms of their age. McDill and

Braze (2000) and Crowe et al. (2003) found that the age-class distribution of a forest is a major factor in determining how difficult harvest scheduling models with adjacency constraints are to solve. Following McDill and Braze (2000), four different initial age-class distributions were used in creating the forests for this study. The initial age-class distributions specify target proportions of the total forest area in each age class representing 1) immature, 2) regulated, 3) mature, and 4) old-growth structures (Table 2). The actual age-class distributions for individual forests can vary slightly from the distributions in Table 2 since an entire stand must be assigned to one age-class.

Table 2: Target distribution of area by age-class for the four initial age-class distributions.

	Percent of total area by age class			
Age class	Immature	Regulated	Mature	Old-growth
1-20	35	25	10	—
21-40	30	25	15	—
41-60	20	25	20	—
61-80	15	25	25	—
81-100	—	—	30	100

Table 3: Model parameters associated with each formulated forest planning problem.

Parameter	Value
Number of planning periods	5 periods (100 yrs)
Minimum rotation	4 periods (80 yrs)
Discount rate	4.00%
Regeneration costs	\$391.51/ha
Timber sale costs	\$239.25/ha
Upper bound on harvest increases	10%
Lower bound on harvest decreases	1%
Wood price	\$105.15/m^3
Minimum average ending age	40 yrs
Maximum harvest opening size	50 ha
Minimum age difference for stands harvested together	2 periods (40 yrs)

Problems were formulated using the following Model I structure (Johnson and Scheurman 1977) and model parameters given in Table 3:

$$MaxZ = \sum_{m=1}^{M}\sum_{t=1}^{T} c_{mt}\cdot A_m \cdot X_{mt} \quad (1)$$

Subject to

$$\sum_{t=0}^{T} X_{mt} \leq 1 \text{for m = 1, 2,...,M} \quad (2)$$

$$\sum_{m=1}^{M} v_{mt}\cdot A_m \cdot X_{mt} - H_t = 0 \text{for t = 1, 2,...T ()} \quad (3)$$

$$b_{l,t}H_t - H_{t+1} \leq 0 \text{for t = 1, 2,...T – 1} \quad (4)$$

$$-b_{h,t}H_t + H_{t+1} \leq 0 \text{for t = 1, 2,...T – 1} \quad (5)$$

$$\sum_{m=1}^{M}\sum_{t=1}^{T}\left(Age_{mt}^{T} - \overline{Age}^{T}\right) A_m \cdot X_{mt} \geq 0 \quad (6)$$

$$X_{mt} \in \{0,1\} \text{ for m = 1, 2,...,M and for t = 1, 2,...T} \quad (7)$$

where X_{mt} = a binary variable whose value is 1 if management unit m is selected to be harvested in period t for t = 1, 2,...T; when t = 0, the value of the binary variable is 1 if management unit m is not harvested at all during the planning horizon,

M = the number of management units in the forest,

T = the number of periods in the planning horizon,

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

A_m = the area of management unit m in hectares,

v_{mt} = the sawtimber volume in m^3 per hectare harvested from management unit m if it is harvested in period t,

H_t = the total sawtimber volume in m^3 harvested in period t,

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

$b_{h,t}$ = an upper bound on increases in the harvest level between periods t and t + 1,

Age_{mt}^{T} = the age of management unit m at the end of the planning horizon if it is harvested in period t, and

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

The objective function (1) maximizes the discounted net revenue for the forest across the planning horizon. Logical constraints (2) require that a management unit be assigned to at most one prescription, including a do-nothing prescription. Constraints (3) are harvest accounting constraints. These constraints sum up the harvest volume for each period and assign it to the harvest

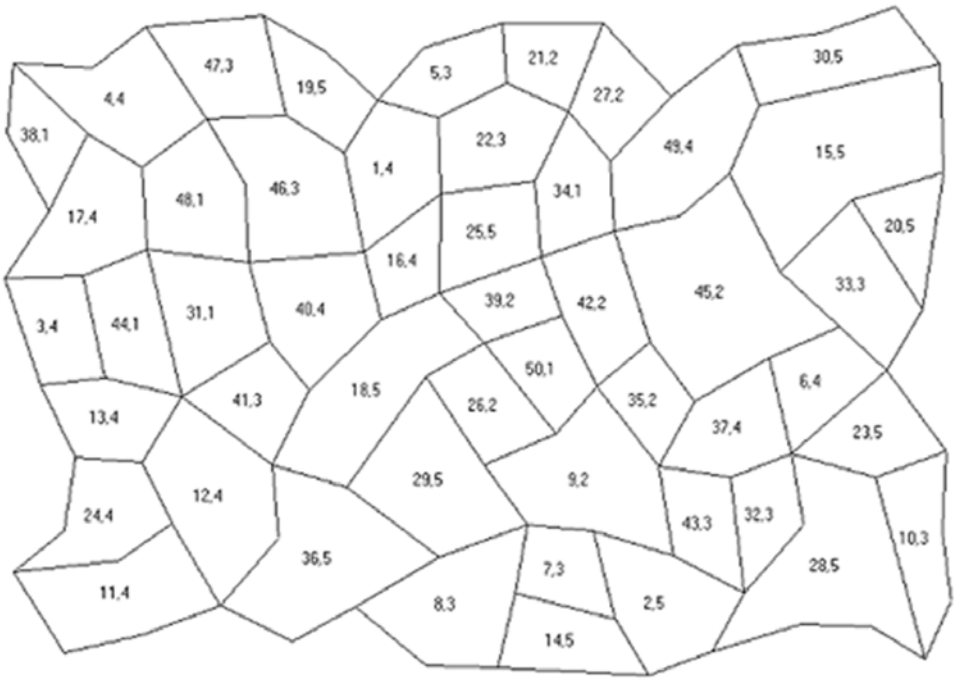

Figure 1: Example 50-stand hypothetical forest where the polygon labels in each stand represent the unit id and initial age-class.

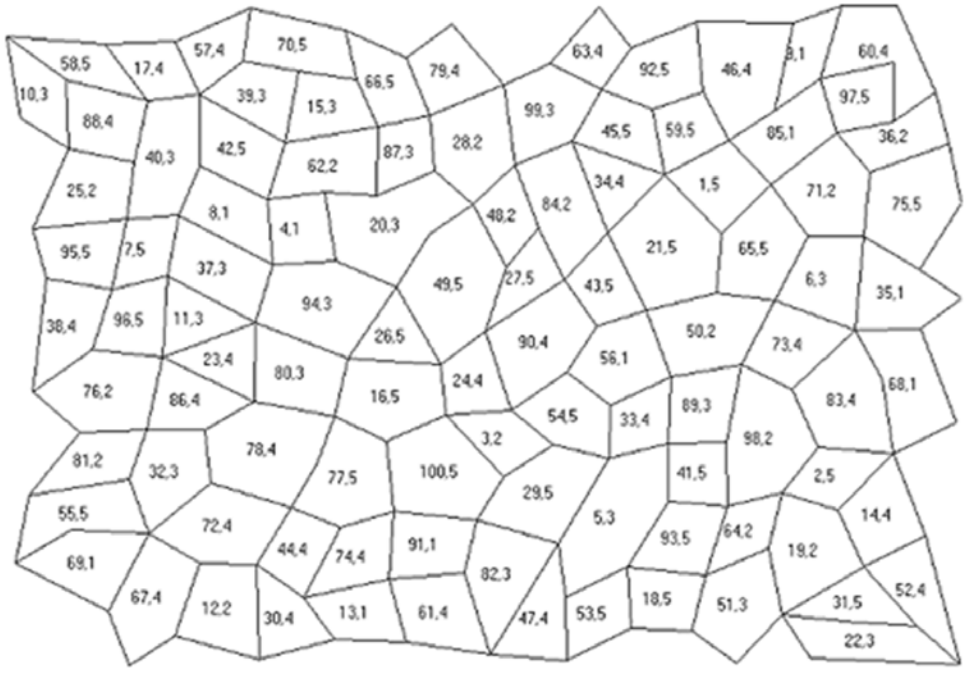

Figure 2: Example 100-stand hypothetical forest where the polygon labels in each stand represent the unit id and initial age-class.

accounting variables (H_t). Constraint sets (4) and (5) impose harvest flow restrictions. The ending age constraint (6) requires the average age of the forest at the end of the planning horizon be at least $\overline{Age}^T$ to prevent the model from over-harvesting the forest. Finally, constraint (7) indentifies the management unit treatment variables as binary.

Four different adjacency constraint types were used in formulating the harvest scheduling models: two URM adjacency constraints (Pairwise and Maximal Clique) and two ARM adjacency formulations (Path and Generalized Management Unit). Pairwise constraints require a single constraint for each pair of adjacent stands for each period (McDill and Braze 2000). Maximal Clique adjacency constraints require a single constraint for each group of mutually adjacent management units rather than each pair (McDill and Braze 2000). The two URM adjacency constraints were formulated as follows:

$$\sum_{m \in C_i} X_{mt} \leq 1 \text{for all } C_j \text{ and } t = 1, 2, ...T \qquad (8)$$

where C_j equals the set of indexes corresponding to the j^{th} pair of adjacent management units for the Pairwise adjacency constraints or the j^{th}set of mutually adjacent management units for the Maximal Clique adjacency constraints (McDill et al. 2002).

Path adjacency constraints and the Generalized Management Unit (GMU) formulation are ARM models that impose only the adjacency restrictions that are needed to preclude harvest openings that would exceed some maximum harvest area (McDill et al. 2002). The maximum harvest area used in this research was 50 ha (Table 3). Path adjacency constraints are based on contiguous sets of management units whose areas minimally exceed the maximum harvest area so that if any management unit is removed from the set the maximum harvest area constraint will be satisfied. These sets were called "paths" by (McDill et al. 2002). The constraints are formulated as follows:

$$\sum_{U \in P_i} X_{Ut} \leq n_{P_i} - 1 \forall P_i \text{ and } t = 1, 2, ...T \qquad (9)$$

where $n_{P_i}$is the number of management units in path i and $P_i$is the set of indexes corresponding to the management units in path i (McDill et al. 2002). Goycoolea et al. (2009) proposed several ways to tighten the Path constraint formulation and an alternative, more efficient algorithm to construct the clusters. However, in this research the original Path algorithm was used and the constraints where not tightened.

The GMU formulation creates variables for contiguous combinations of management units whose combined area does not exceed the maximum allowable harvest area. A maximum age difference among these groups of contiguous stands of two periods was used in this research (McDill et al. 2002, Table 3). These groups of management units are referred to as Generalized Management Units in McDill et al. (2002). In the GMU formulation, constraint set (2) is replaced by the following:

$$\sum_{u \in G_m} \sum_{t=0}^{T} X_{ut} \leq 1 \text{for m} = 1, 2, ...M_o \qquad (10)$$

where M_o is the complete set of original management units and G_m is the subset of management units in M (which includes both M_o and the GMU's) that includes original management unit m (McDill et al. 2002). Also, constraint sets (8) must be included as either Pairwise or Maximal Clique adjacency constraints to complete the ARM using the GMU approach (McDill et al. 2002).

With 160 forests and four adjacency formulations, there were 640 problem instances. The problems were solved and tuned as sets of 20, where a set is defined as a given forest size, initial age-class distribution, and adjacency formulation type. The problem sets were first solved using the Cplex§11.2 interactive optimizer's default parameters to a gap of 0.1% and with node files stored on the hard drive. Problems sets were tuned with the gap and node file parameters fixed, and then solved again with the suggested parameter settings from the tuning tool. The average solution time was calculated for each problem set using the default and tuned parameter settings. One-tailed paired t-tests were conducted using R 2.7.1 (R Development Core Team 2008) to identify significant differences ($\alpha = 0.05$) between the mean solution time with the default and tuned parameter settings. The Anderson-Darling test for normality was used to assess the assumption of normality with small samples. P-values were greater than 0.05 for the mean differences in each problem set thus failing to reject the null hypothesis that the differences follow a normal distribution. It would be desirable to have a single optimal parameter set that would work well for all spatially-explicit forest planning models. To see if this was possible, three problems from each set were randomly selected to create a mixed problem set with 96 problems. These problems were solved, tuned, and resolved using the suggested parameter settings. All work was done on a computer with an Intel§Xeon 3.73 GHz dual processor with 3.0 gigabytes of RAM.

4 Results

The results from the 50-stand and 100-stand forest problem sets are presented in Tables 4 and 5. With few exceptions, the mean solution times were smaller when the problem sets were solved with the suggested parameter settings from the performance tuning tool than with

Table 4: Results for the 50-stand forest problem sets.

Constraint type	Default parameter mean sol. time (sec)*	Tune parameter mean sol. time (sec)*	Tuning time (sec)	t-stat	p-value
		Immature forests			
Pairwise	0.8 (0.12)	0.7 (0.07)	54.4	0.736	0.235
Maximal Clique	0.6 (0.07)	0.5 (0.08)	46.7	1.233	0.116
Path	1.2 (0.25)	0.9 (0.11)	71.2	1.291	0.106
GMU	3.3 (0.91)	1.7 (0.27)	1,059.0	2.165	0.022
		Regulated forests			
Pairwise	2.3 (0.99)	1.0 (0.33)	598.4	1.496	0.076
Maximal Clique	1.3 (0.28)	1.3 (0.64)	561.2	-0.110	0.450
Path	3.7 (1.51)	1.8 (0.42)	995.7	1.510	0.074
GMU	33.8 (17.73)	12.0 (3.61)	7,363.5	1.292	0.110
		Mature forests			
Pairwise	231.7 (65.32)	214.3 (77.15)	103,830.1	0.393	0.349
Maximal Clique	169.5 (35.40)	215.6 (79.35)	91,060.4	-0.673	0.255
Path	1,238.3 (651.44)	340.3 (189.76)	254,943.5	1.478	0.078
GMU	1,335.1 (440.38)	1,201.8 (384.70)	426,756.3	1.088	0.145
		Old Growth forests			
Pairwise	806.2 (180.10)	231.6 (61.96)	194,482.7	3.120	0.003
Maximal Clique	554.1 (101.61)	152.1 (29.04)	147,728.4	3.604	0.001
Path	317.0 (54.60)	66.8 (20.87)	77,030.7	4.617	0.000
GMU	2,192.4 (288.90)	726.5 (180.11)	563,463.6	4.168	0.003

*Standard error (SE) in parenthesis.

Table 5: Results for the 100-stand forest problem sets.

Constraint type	Default parameter mean sol. time (sec)*	Tune parameter mean sol. time (sec)*	Tuning time (sec)	t-stat	p-value
		Immature forests			
Pairwise	6.4 (1.67)	2.0 (0.46)	1,540.3	2.420	0.013
Maximal Clique	6.3 (1.53)	1.9 (0.43)	2,672.8	2.955	0.004
Path	2.6 (0.70)	1.9 (0.92)	741.9	0.589	0.281
GMU	13.5 (5.24)	2.7 (0.34)	2,851.0	2.039	0.028
		Regulated forests			
Pairwise	3.9 (1.40)	1.0 (0.20)	839.4	2.265	0.018
Maximal Clique	4.4 (2.57)	0.8 (0.20)	899.5	1.415	0.087
Path	2.1 (0.55)	4.3 (2.38)	923.6	-1.155	0.131
GMU	14.2 (5.84)	4.2 (1.35)	2,998.7	2.000	0.030
		Mature forests			
Pairwise	1,207.3 (275.27)	1,159.9 (553.71)	390,398.5	0.098	0.461
Maximal Clique	1,417.3 (484.00)	835.0 (315.30)	387,729.9	1.119	0.139
Path	112.6 (46.83)	14.5 (3.35)	17,644.5	2.101	0.025
GMU	6,150.0 (3,420.51)	447.5 (190.55)	214,619.0	1.711	0.052
		Old Growth forests			
Pairwise	112.2 (22.77)	32.2 (6.54)	29,086.8	3.213	0.002
Maximal Clique	76.2 (17.08)	31.0 (4.32)	25,790.5	2.452	0.012
Path	53.3 (11.64)	18.7 (2.61)	15,308.5	2.679	0.007
GMU	1,327.4 (199.47)	172.1 (34.72)	238,726.5	5.870	0.000

*Standard error (SE) in parenthesis.

the default parameter settings. Exceptions were regulated and mature forest problems formulated with Maximal Clique adjacency constraints for the 50-stand problems and the regulated forest problems with Path adjacency constraints for the 100-stand problems. However, for the 50-stand problems, these decreases in mean solution time were only statistically significant for immature forests with GMU adjacency constraints and old-growth forests (Table 4). On the other hand, the differences were statistically significant for 10 of the 16 large forest problem sets (Table 5). Overall, using tuned parameters reduced solution times by an average of 39% for the 50-stand problems and 55% for the 100-stand problems.

Tables 4 and 5 also show the time it took the performance tuning tool in Cplex§to determine the optimal parameter settings for each problem set. Tuning times were almost perfectly correlated with solution times, which were, as expected, greater for mature and old-growth forest problem sets compared to the immature and regulated forest sets (Table 4 and Table 5). Fifty-stand old-growth forests with GMU adjacency constraints took the longest time to tune among the problem sets at over 156.5 hrs (Table 4). This reflects the somewhat surprising result that for the old-growth forests, the 50-stand problems took two to seven times as long to solve compared with the 100-stand problems.

The most frequent parameter settings resulting from tuning the problem sets are shown in Table 6.

Table 6: Most frequent recommended parameter settings for Cplex§among all problem sets from the performance tuning sessions.

Parameter name	Recommended value	Occurrence (Number of problem sets)
mip cuts flowcovers	1	20
mip cuts mircut	1	20
mip limits cutpasses	1	9
mip strategy probe	-1	10
mip strategy heuristicfreq	50	17
mip strategy variableselect	4	19

The mean solution time for solving a randomly selected mixed set of 96 problems with the default parameters was 255.5 (SE = 69.9) seconds. In contrast, the mean solution time using the tuned parameter settings was 144.0 (SE = 46.1) seconds. This 44% reduction in mean solution time was significant (p-value = 0.048) at the 5% level. The tuning time for the 96 problems was 604,128 seconds. The recommended parameter settings from tuning this set included only mip limits cutsfactor 30 and mip strategy rinsheur 100.

The solutions obtained for a given similar problem (forest size and age-class) varied among adjacency constraint types. As one would expect, objective function values, solved to the specified 0.1% gap, increased when problems were formulated with the ARM adjacency constraint types compared to the URM adjacency constraint types. Sometimes the difference was substantial, especially with forests with mixed age-class distributions. For example, a 50-stand immature forest formulated with Maximal Clique adjacency constraints had an objective function value of $183,025.95 while the same problem formulated with Path adjacency constraints had an objective function value of $227,049.86. Another example was an overmature 100-stand forest, which had an objective function value of $774,101.31 with Pairwise adjacency constraints compared to $781,287.89 when formulated with GMU constraints. These results were not observed, however, with forests with an initial old-growth age-class distribution. These problems had similar objective values among all adjacency constraint types with between $5.00-$400.00 differences in objective values and no clear pattern between URM and ARM formulations. Tuning resulted in no clear pattern in objective values of problems solved with the default and tuned parameters.

5 Discussion

This article addresses an approach to improving the efficiency of solving harvest scheduling models with adjacency constraints using the BBA. The mean solution times for sets of similarly formulated forest planning problems were reduced when the problems were solved with the recommended parameter settings from the tuning tool compared with the default parameter settings in Ilog's Cplex§11.2 optimizer. Exceptions where the mean solution time was not decreased by tuning the parameter settings occurred in the 50-stand regulated and mature forests with maximal clique constraints and 100 stand regulated forests with path constraints. The mean solution time was significantly lower using the tuned parameter settings compared to the default settings for most sets of formulated problems. Specifcally, solution time reductions were significant for problem sets with 50 and 100-stand forests with old-growth age-class distributions across all adjacency constraint formulations. Certain parameters were recommended from the tuning tool more frequently than others, although they were not consistent among problem sets (Table 6). The majority related to how cuts are performed during the BBA. Moderately applying cuts, specifically flow covers and MIR

cuts, and limiting the number of cutting plane passes were often beneficial in improving solution time. Also, probing on variables prior to branching was typically not recommended, suggesting that an investment in probing does not produce gains in solution time for these problems. The tuning tool also suggested that performing the node heuristic at short intervals, every 3^{rd} node, particularly on mature and old-growth forests, reduces the solution time for these types of forests with Cplex§. However, solving the large subset of problems suggested that applying the RINS heuristic at a larger interval was a more reliable strategy for a variety of MIP forest planning problems. Ultimately, this suggests that some kind of periodic heuristic has a positive effect on improving the efficiency of MIP forest planning problems.

For forests with immature, regulated and overmature age-class distributions, substantial objective function value increases were observed when forest planning problems were formulated with ARM adjacency constraints as compared to URM adjacency constraints. Similar results were shown in McDill et al. (2002). However, the old-growth forest formulations did not exhibit the same pattern as the other age-class distributions. This is not surprising since in these forests all stands have the same initial age class; the model is more-or-less indifferent about which stands are harvested in a given period, so there is little benefit to clustering stands into larger harvest blocks. This is a somewhat artificial result, since in actual old-growth forests, there will be heterogeneity between stands that likely would lead to gains from using an ARM approach rather than a URM approach.

This research illustrates three key findings. The first is that changes in parameter settings in off-the-shelf optimizers can potentially have a significant effect on the time it takes to solve harvest scheduling models with adjacency constraints. Prior to this work, little research has focused on exploring these parameters and analyzing ways to determine optimal settings for given problem types. Second, tuning is more costly in terms of time for harder problems, but the pay-off is also generally larger for these problems. Problem sets that are relatively easy to solve, e.g., those for smaller forests with younger age-class distributions, generally take less than an hour to tune, but they are easy to solve in any case, so there is little gain from tuning. Conversely, harder problems required many hours, if not days, to tune. But the gains from tuning tend to be more significant with harder problems. For instance, the gains from tuning were more frequently significant for the larger forest problems than for the smaller forests, and they were always highly significant (at least at the 0.001 level) for all the old-growth problems, regardless of forest size. In addition, the results here confirm that tuning takes at least 6-8 times longer, as reported by Ilog (ILOG 2008), than solving a given problem and in most cases substantially longer. On average, our problems took over 200 times as long to tune as it did to solve one problem. This time investment is realized, however, in significantly lower solution times using the suggested parameter settings. Thus, the investment in tuning solution parameters is only likely to be worthwhile when similar problems will be solved many times. The third key finding is that the suggested parameters associated with a given problem type resulting from the tuning utility are specific to certain types of problems. This is important practically because each type of problem, or sets of similar problems, would have to be tuned separately to determine the optimal parameters.

In conclusion, this research provides insight into the value of determining the optimal parameter settings for specific types of harvest scheduling problem with adjacency constraints. Even though this work was based on relatively simple hypothetical forests, one would expect similar results with real forests based on similar age-class distributions and size.

Acknowledgements

This research was supported by funding from the Penn State School of Forest Resources and the Pennsylvania Department of Conservation and Natural Resources Bureau of Forestry.

References

Barahona, F., R. Epstein, and A. Weintraub. 1992. Habitat dispersion in forest planning and the stable set problem. Oper. Res.40(1): S14–S20.

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

Bettinger, P., K. Boston, and J. Sessions. 1999. Combinatorial optimization of elk habitat and timber harvest volume. Env. Mod. and Ass. 4(2–3): 143–153.

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

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

Caro, F., M. Constantino, I. Martins, and A. Weintraub. 2003. A 2-Opt tabu search procedure for the

multi-period forest harvesting problem with adjacency, green-up, old growth and even flow constraints. For. Sci.49(5): 738–751.

Constantino, M., I. Martins, and J. Borges. 2006. A new mixed integer programming model for harvest scheduling subject to maximum area restrictions. Oper. Res. 56(3): 542–551.

Crowe, K., J. Nelson, and M. Boyland. 2003. Solving the area restricted harvest scheduling model using the branch and bound algorithm. Can. J. For. Res. 33(9): 1804–1814.

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

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

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

ILOG. 2008. CPLEX 11.2 user's manual. ILOG, Inc., Incline Village, Nevada.

Johnson, K.N., and H.L. Scheurman. 1977. Techniques for prescribing optimal timber harvest and investment under different objective – discussion and synthesis. For. Sci. Monogr. 18.

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

McDill, M.E., and J. Braze. 2000. Comparing adjacency constraint formulations for randomly generated forest planning problems with four age class distributions. For. Sci. 46(3): 423–436.

McDill, M.E., and J. Braze. 2001. Using the branch and bound algorithm to solve forest planning problems with adjacency constraints. For. Sci. 47(3): 403–418.

McDill, M.E., S.A. Rebain, and J. Braze. 2002. Harvest scheduling with area-based adjacency constraints. For. Sci. 48(4): 631–642.

Murray, A.T. 1999. Spatial restrictions in harvest scheduling. For. Sci. 45(1):1-8.

Murray, A.T., and R.L. Church. 1995. Heuristic solution approaches to operational forest planning problems. OR Spekt. 17: 193–203.

Murray, A.T., and A. Weintraub. 2002. Scale and unit specification influences in harvest scheduling with maximum area restrictions. For. Sci. 48(4): 779–788.

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

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

R Development Core Team. 2008. R: Language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria.

Richards, E.W., and E.A. Gunn. 2000. A model and tabu search method to optimize stand harvest and road construction schedules. For. Sci. 46(2): 188–203.

Thompson, E.F., B.G. Halterman, T.S. Lyon, and R.L. Miller. 1973. Integrating timber and wildlife management planning. For. Chron. 47: 247–250.

Weintraub, A., G. Jones, A. Magendzo, M. Meacham, and M. Kirby. 1994. A heuristic system to solve mixed integer forest planning models. Oper. Res. 42(6): 1010–1024.

Williams, H.P. 1993. Model building in mathematical programming. Ed. 4. Wiley, New York. 356 p.