SPATIAL FOREST PLANNING ON INDUSTRIAL LAND: A PROBLEM IN COMBINATORIAL OPTIMIZATION

Karl R. Walters[1]

ABSTRACT.—Recently, member companies of the American Forest & Paper Association adopted guidelines for the management of their lands that impose significant challenges to harvest scheduling. In particular, foresters face difficulties not only in scheduling harvest blocks, but even in delineating them. The spatial forest-planning problem (SFPP) involves simultaneous allocation and scheduling of cells to form harvest blocks that meet adjacency, green-up and opening-size constraints. A preliminary investigation of simulated annealing and tabu search was made to test their suitability to the SFPP. Simulated annealing yielded better solutions and shorter execution times for a number of contrived but otherwise representative problems.

INTRODUCTION

Harvest scheduling is a traditional exercise that has been carried out by forest land managers on public and private lands alike. Industrial forest management has generally been distinguished by a much higher rate of intensification than has been the case on public lands. Whereas public land management has characteristically balanced timber and non-commodity forest production, industrial plantation establishment has been dominated by profit maximization and cost reduction through economies of scale. The result is a patchwork of large, uniform even-aged plantations of very high productivity that have been able to meet society's demand for high quality and affordable forest products. Unfortunately, the same public that demands forest products at low cost has made it clear to the forest industry that it does not favor the large clearcut areas that have been characteristic of southern pine plantations. In response, member companies of the American Forest and Paper Association (AF&PA) have agreed to comply with the Sustainable Forestry Principles and Implementation Guidelines; other companies have adopted their own set of guidelines to promote sustainable forestry. These new principles place renewed emphasis on non-timber resources and require that harvest operations be scheduled according to spatially specific objectives that call for smaller scale harvest operations and diversification of harvests over the landscape. Forest companies are now facing the challenge of scheduling spatially specific timber management activities over time to achieve their traditional objectives and the new environmental goals.

The Nature of the Problem

Champion International is a large forest-products company with land holdings throughout the United States with significant acreage throughout the Southeast and into Texas. Like most other companies in the region, the bulk of these forestlands have been managed as southern pine plantations. These plantations are typically large (several hundred to upwards of several thousand acres) and have been harvested multiple times with rotations ranging from 10 to 30 years, depending on growth rates. Harvest scheduling has typically been done based on maximization of present net worth and land expectation value, and concentrating harvesting operations near one another has been used to minimize harvest costs. The resulting landscape is generally one of overwhelming uniformity, with large tracts of the same species of very similar age.

Champion's new Sustainable Forestry Initiative (SFI) is based on the AF&PA guidelines, but operating guidelines are at least as strict and sometimes more restrictive than those set out by AF&PA. For example, clearcut harvesting is limited to areas less than 240 acres and only where necessary will clearcut areas exceed 120 acres. Clearcut

[1] Karl Walters is a graduate research assistant at the College of Forest Resources, North Carolina State University, Raleigh, NC 27695.

harvest areas are considered *contemporary* if they are established within a fixed number of years of one another. Until better estimates of regeneration response are known, Champion has conservatively set the *green-up period* at 4 or 5 years in most regions. Clearcutting will not be permitted within 300 feet of a contemporary clearcut harvest area unless a watercourse that requires more restrictive riparian zone management separates the two.

Although it is quite easy to state that forest management planning as an activity is deciding what activities to implement, in what place and at what time, the actual process of making these determinations is far from easy. In years past, locating a harvest activity had little consequence elsewhere in the forest or in future years, except for the fact that the harvested timber was no longer readily available. In modern planning situations, the location of a harvest activity can have impacts on large areas of the forest by making them unavailable for harvesting, not only in the current planning period, but for future ones as well. A poor choice made in past years can severely limit options in future periods. In the case of Champion and other companies operating in the southeast, past practices are probably the biggest obstacles to overcome under the new operating guidelines. In general, making harvest blocks as large as possible and keeping green-up delays as short as possible is the best way to mitigate adjacency constraints.

Champion is not unique with its SFI and other companies in the southeast have adopted similar operating principles. In fact, the majority of forest companies implementing these new types of operating guidelines face similar difficulties in implementing them. They require some method of determining spatial harvest schedules in a timely manner, and they have to contend with a major shift in harvesting logistics associated with smaller harvest areas dispersed over much wider areas. The purpose of this paper is to discuss some alternative strategies for solving spatially constrained harvest schedules and the kinds of results to expect.

Literature Review

Linear programming has long been the primary method for developing long-term forest management schedules. As the need for spatial feasibility in these schedules became more common, it is only natural that researchers tried to incorporate spatial constraints within their preferred modeling framework. For example, we have seen the development of aggregate emphasis and coordinated allocation choice formulations within FORPLAN, but these formulations assume previously defined activities schedules and layouts (Connelly 1988). For a single watershed, developing one or two spatially feasible activity schedules is possible, but for an entire forest this task is not feasible. Furthermore, there is no guarantee that such schedules are even close to optimal given the exceedingly large number of alternative layouts that are possible.

Meneghin et al. (1988) demonstrated methods of efficiently coding adjacency constraints within a mixed-integer programming formulation. The rationale for this approach is to limit the number of constraints in the integer formulation to fit within the capabilities of commercial IP codes: if a given problem can be correctly represented by fewer constraints, then a correspondingly larger problem should be solvable. This paper spawned many other papers seeking even more efficient ways of structuring adjacency constraints (Torres-Rojo and Brodie 1990; Jones et al. 1991; Yoshimoto and Brodie 1994). Murray and Church (1995a) compared many of these different approaches and found that structures that yielded the fewest constraints increased computational effort rather than decreasing it. Snyder and ReVelle (1996) used derivations of the pairwise methods of Meneghin et al. (1988) to solve spatial harvest schedules using simplex with limited branching and bounding. All of these approaches assume that harvest blocks are previously delineated and that the adjacency restrictions prevent adjacent blocks from being harvested simultaneously.

O’Hara et al. (1989), Clements et al (1990) and Nelson and Brodie (1990) all developed mixed-integer harvesting formulations with pairwise harvest adjacency constraints and none used linear programming with branch and bound to find solutions. O’Hara et al. used a biased sampling search technique and the latter two used Monte Carlo Integer Programming (MCIP). Although it required previously defined harvest blocks, the BLOCK heuristic presented by Clements et al. (1990) differed from the other heuristic approaches in that it only prevented adjacent blocks from being harvested simultaneously when the opening created was too large. This type of formulation has not been demonstrated in LP-based systems.

Lockwood and Moore (1993) gave an example application of spatial harvest scheduling using simulated annealing (SA) where each stand in the forest could be considered for a single treatment over the planning horizon but a large number of stands were involved. To make harvesting economically feasible, the algorithm is forced to aggregate adjacent stands into harvest blocks, assigning penalties to instances where blocks are too small or too large. Similar types of penalties are applied to prevent adjacent stands from being harvested within the green-up period. Murray and Church (1995b) used three different heuristic solution methods (interchange, SA and tabu search (TS)) to solve spatially constrained harvest schedules that also included road construction decisions and they reported near optimal results for short planning horizons.

Remsoft Inc. (1996) has developed a spatial forest planning system based on a two-tier approach linking strategic and tactical planning. The system uses linear programming to develop a strategic harvest schedule that reflects long term goals such as harvest flows, silvicultural costs, forest structure requirements and so forth. The strategic harvest schedule is then used to guide a blocking and scheduling algorithm (Stanley) that uses local improvement and random restart heuristics to generate spatially feasible harvest schedules for a shorter tactical planning horizon. The system has been applied to numerous public land and industrial forest planning situations with good success.

In relation to the spatial forest planning problem (SFPP) faced by Champion International and other industrial forestland owners, the methodologies outlined have major drawbacks. Foremost is the assumption that one can delineate good harvest block allocations a priori. Remsoft Inc. (1994) compared spatial harvest schedules prepared by Crown land licensees in New Brunswick with schedules prepared using the *Crystal* (Jamnick and Walters, 1993) and *Block* (Clements et al. 1990) algorithms to generate harvest blocks and schedule them under adjacency constraints. Their findings suggested that manually delineated blocks resulted in sustainable harvest levels that were significantly lower than the heuristically obtained schedules, even though both processes used identical forest stratification and yield estimates. This would suggest that at least some of the *losses* associated with spatial constraints are in fact due to poor block configuration.

Although the work on incorporating adjacency constraints within a mixed integer programming formulation have yielded interesting results from a theoretical standpoint, the ability to solve large-scale harvest scheduling with adjacency constraints is still elusive. Unlike the simple adjacency concept used in most papers, the SFI guidelines would require recognition of proximal polygons that are within 300 feet as adjacent polygons, significantly increasing the number of adjacency relationships to contend with.

One of the drawbacks to the Stanley algorithm is its dependence on the strategic harvest schedule to assign harvest-timing choices to forest types. In the absence of adjacency information, an LP-based harvest schedule can easily assign more acres to a harvest prescription in a given period than would be allowed when spatial constraints are applied. For example, suppose the LP assigns all of a forest class to be harvested in a single period. If the class is made up of a single plantation that exceeds the maximum opening size, then it cannot all be harvested in one period. Alternatively, the class may be composed of many stands too small to be harvested on their own economically. Unless all of these stands have eligible neighbors they can be combined with, there is no way to liquidate the entire class according to the LP schedule. Obviously, deviations from the LP schedule are necessary to meet the spatial requirements and the solution may be driven further from true optimality as the scheduling algorithm tries to mitigate the penalties. That said, the Remsoft algorithm generally performs very well.

The SA algorithm presented by Lockwood and Moore (1993) worked with individual stands and to apply this algorithm to the spatial forest-planning problem as outlined, stands need to be subdivided into smaller pieces. Although the authors suggested that subdividing stands might be helpful to mitigate block size problems, they did not pursue it. Increasing the number of allocation units should not pose untoward difficulties, but the algorithm would need to be modified to recognize the difference between adjacent and proximal polygons: only adjacent stands are used to develop harvest blocks, but proximity is important for determining if adjacency restrictions apply.

Combinatorial Nature of the Problem

The SFPP simultaneously allocates polygons to harvest blocks and schedules those blocks for harvest under adjacency constraints. Even if we ignore the scheduling aspect for the moment and consider only a single harvest activity, the problem still grows exponentially in the number of periods to plan for. For example, suppose we wish to know how much of a given plantation can harvested in a single period. If the plantation is composed of 10 cells, there are 2^{10} or 1024 potential arrangements of those 10 cells and if the plantation is composed of 100 cells, there are 1.267×10^{30} combinations. Now, consider that there may tens of thousands of cells and up to 20 harvest periods and the number of alternatives quickly becomes astronomically large. Since enumeration and deterministic algorithms like branch and bound are not feasible for problems of this size, the alternative is to consider heuristics.

Heuristic approaches are based on the idea of making incremental improvements by changing elements of a solution iteratively. Although the SFPP offers a very large number of alternatives, many of them represent infeasible solutions and the feasible region is not a continuous space. If one considers one-at-a-time changes to the solution space, each change represents a neighbor solution and the size of this neighborhood is a function of the number of possible changes. For example, if the planning problem has a planning horizon of 4 periods (plus a no-harvest option), then changing the harvest period of a single cell results in a neighborhood of size 4. If the proposed change involves swapping the harvest period of two cells in the forest, then the neighborhood size is much larger: $n(n-1)/2$ where n is the number of cells in the forest. Computation time for estimating effects of a proposed change to a solution are generally a function of neighborhood size and the larger the neighborhood, the more computation that will be required. If it is computationally expensive to evaluate the objective function, the choice of neighborhood is critical to maintain reasonable run-times.

While neighborhood size is an important consideration in implementing a heuristic approach, the terrain of the neighborhood is also a consideration. Ideally, the neighborhood chosen is neither flat (objective function remains the same over many changes) nor overly steep (objective function changes by large values with each modification to the solution). Because the spatial requirements make many of the alternative solutions infeasible, there are two recourses for finding solutions: either maintain feasibility at all times with significant computational effort or, as is most common, forego feasibility but penalize solutions for violating spatial constraints. Although the use of penalties reduces computational burden, neighborhoods in the SFP problem tend to have spiky neighborhood because of the spatial requirement penalties. Depending on the heuristic being used, these spikes may make it difficult to converge to a good solution

METHODS

For this paper, I considered 2 different heuristic approaches: simulated annealing (SA) and tabu search (TS). Both are really extensions of a local improvement algorithm where changes are proposed and accepted only if they improve the objective function value. The algorithm continues to run until no further improvement is possible. While local improvement algorithms can become trapped at a local optimum, the SA and TS algorithms avoid becoming trapped by allowing changes that lead away from the local optimum. Each of these approaches has yielded good results for different combinatorial optimization problems, but neither is universally superior. The purpose of this paper is to determine the relative advantages and disadvantages of these algorithmic approaches with respect to the SFP.

Simulated Annealing

Numerous references are available that discuss the structure of SA algorithms (Kirkpatrick et al. 1983; Johnson et al. 1989) so I will not go into great detail. Basically, a SA algorithm begins with an initial solution S, and a definition of neighborhood that allows perturbation of a solution. If a perturbation results in an improved objective function value, the move is accepted. If the move does not improve the objective function value (an uphill move), it may still be accepted and the probability of acceptance depends on the magnitude of the change and a temperature parameter (T).

Initially, the value of T is set to a high value such that the probability of accepting an uphill move is similarly high but as the algorithm progresses, the temperature parameter T is gradually lowered such that uphill moves are accepted less and less frequently. Johnson et al. (1989) suggest an initial temperature where about 40% of moves are accepted. Temperature reductions are performed after a fixed number of moves and the algorithm terminates when the number of moves accepted falls below a threshold value and no improvements to the objective function have after a fixed number of moves. The SA algorithm used in this study did not incorporate any special modifications and conformed to the suggested structure of Johnson et al. (1989).

Table 1.—Parameter settings for simulated annealing algorithm

Parameter	Value	Purpose
L	20 * polygons in forest	Determines the number of perturbations to attempt before reducing the temperature parameter
Tfactor	0.95	Multiply T by tfactor to reduce T after L moves attempted
Minpercent	0.02	Threshold for determining stopping criteria; if no improvement in objective function value after 10 temperature reductions, STOP
Initprob	0.40	The initial temperature is varied up and down in an abortive annealing run to determine a temperature at which about 40% of the moves are accepted.

Tabu Search

Rather than selecting one neighboring solution and deciding to implement the move or not as is done in SA, a TS algorithm evaluates all of the neighboring solutions and implements the move that improves the objective function value most. If all of the moves are uphill moves then the TS implements the move that reduces the objective function value by the smallest amount. Although these occasional uphill moves provide a means for escaping local optima, a mechanism is required to prevent the algorithm from immediately returning to the previous value when that neighborhood is revisited next time.

The key feature of TS is the use of short-term memory to guide searching the solution space (Glover 1990). To better understand this metaphor, consider branch and bound to be a long-term memory algorithm: once a node in a branch and bound tree has been fathomed, the algorithm *remembers* this and never returns to it. On the other hand, SA is a memory-less algorithm because its traversal of the solution space is completely random, and it may make the same move many times over the course of an annealing run. In tabu search, once a move has been accepted, that move is made taboo for a period of time to force the algorithm to explore other parts of the solution space. The period of time a move is taboo is called the tabu tenure.

Some other features of TS are the concepts of aspiration criteria and diversification. Basically, the idea behind aspiration criteria is that occasional tabu moves may be allowed if they advance to a more desirable solution. Commonly, aspiration criteria are defined as moves that yield a better solution than any found thus far, and when the TS encounters such a move, the aspiration criteria override the tabu status and allow the move to proceed anyway. Diversification is used when improvements in objective function value become too infrequent and a change is made simply to cause the algorithm to search another part of the solution space in hopes of finding better solutions. Diversification may include complete restarts with a new random solution, or some larger scale perturbation of the current or candidate solution. The algorithm terminates when a fixed number of diversification moves are made without improving the objective function value. The TS algorithm used in this study basically follows the suggestions of Glover (1990); diversification was accomplished by randomly changing the harvest periods of cells within the candidate solution.

Table 2.—Parameter settings for tabu search algorithm

Parameter	Value	Purpose
Tabu tenure	k periods where k=#polygons in forest	Determines the length of time (iterations) a move is not allowed
Diversification count	20	Algorithm terminates after 20 diversification moves.

RESULTS

To make the comparisons between SA and TS fair, both algorithms employed the same evaluation routines. The user specifies the minimum and maximum block size as well as the length of green-up period required between adjacent harvest blocks. To prevent violations of the spatial constraints, penalties are applied at the rate of 2000 units per block-adjacency violation and 200 units per size violation. If even flow constraints are applied, the objective function is calculated as: *Z= MIN(periodic harvest)+0.2*(total harvest)-0.1*(range in harvest levels)-adjacency_penalty-size_ penalty*. Without even flow constraints, the objective function is simply *Z=(total harvest)-adjacency_penalty-size_penalty*.

To test how well these algorithms perform, I set up a number of small test problems to measure quality of solution and execution time. Each algorithm was run 10 times for each problem instance with the best and average performance noted.

Plantation Liquidation Problem

The plantation liquidation problem considers the problem of a plantation that was established with the idea of liquidating it in one harvest. Given the new operating guidelines in place, how much of the plantation can be liquidated in a single pass and how long does it take to liquidate the entire plantation with economically viable harvest blocks? To model the latter, I assumed a base value of 100 for cells cut in the first year, no growth in future period and a 4% discount rate. Although plantations in the south will most certainly continue to grow beyond their economic rotation age, higher discount rates will depreciate their value more quickly. The results in Table 1 include 4 different sized problems where the single pass liquidation was used and a single SA was able to find the optimal solution in every instance of the 7x7 and 5x10 problem sets and came very close to the optimal each time in the 10x10 and 5x20 problem sets. The TS algorithm fared less well but was still able to find the optimal solution at least once in 10 runs for each of the smaller problem sets. On the larger problems, the TS algorithm performed significantly poorer than the SA algorithm. For a single run on a 25x25 problem set, SA produced what appears to be a rational means of solving the problem: make blocks as large as possible separating them by a minimum width buffer; Additional cells are left unharvested to avoid exceeding the maximum opening size.

Table 3.—Results of applying SA and TS algorithms to the single pass plantation liquidation problem

	Simulated Annealing Results			Tabu Search Results			
Problem	Best	Average	Average # Iterations	Best	Average	Average # Iterations	Optimal Solution Value
7x7	21000	21000	37600	21000	19200	21400	21000
5x10	22500	22500	37800	22500	21050	22300	22500
10x10	40500	40100	222800	37000	35200	51300	41000
5x20	42500	41800	78200	37500	35400	31100	unknown

Table 4.—Results of applying SA and TS algorithms to the problem of complete plantation liquidation

Simulated Annealing Results				Tabu Search Results			
Best Solution	Average	Average # Iterations	Max./Min periods req.	Best Solution	Average	Average # Iterations	Max/Min periods req.
48475	47725	179800	12/5	42708	35683	222600	13/10

The solutions to the second liquidation problem were more varied, but overall the algorithms tried to make period 1 blocks as large as possible while creating harvest blocks to be harvested in subsequent periods in the buffer areas. If these blocks are unacceptable, it may be necessary to place constraints on block shape to discourage *tentacles*.

Harvest Scheduling Problem

Although the liquidation problem is an interesting stand-level question, it has limited applicability in a forest-wide context. A more relevant question is to what happens when even flow constraints must also be satisfied. Unlike linear programming solutions that allow infinite divisibility of decision variables, the SFPP forces outputs to be produced in discrete quantities. How well the algorithm is able to balance flows is a function of the size of the problem formulation and the relative effect of a single cell on forest-wide output levels. With a small forest, it may be quite difficult to achieve even the semblance of even-flow whereas on large forests, even-flow can be easily attained.

Table 5.—Results of applying SA and TS algorithms to different 100 cell forests with even-flow constraints

	Simulated Annealing Results			Tabu Search Results		
	Best Solution	Average	Average # Iterations	Best Solution	Average	Average # Iterations
4 types in 4 stands	31332	28655	154400	9020	6922	321800
4 types, random	67060	58232	167600	8922	7896	361500
1 type	16538	15673	132400	8820	7960	244000

Again, SA did markedly better than TS in both execution time and quality of solution. In fact, TS yielded exceedingly poor results for the 4-stand and random-stand problems. However, with the uniform forest type, the TS algorithm performed almost as well as the SA. Harvest flows produced using SA were very good for the uniform forest (+/- 4%) and less acceptable for the multi stand cases (+72/-42% for random, +30/-19% for 4-stand).

DISCUSSION

Neither algorithm was implemented for use outside of this study and overall performance of the programs could be improved dramatically. Both algorithms used the same evaluation procedures and improving the efficiency of these procedures would speed up both programs. Since the complete evaluation routine is much more computationally intensive, any improvements in its efficiency would be of greater benefit to the TS algorithm that calls that procedure multiple times during execution. Nevertheless, the simulated annealing algorithm appeared to be a better performer overall, not only in terms of execution time but also in the quality of solutions.

I suspect the poorer performance of the tabu search is due to the spiky neighborhood referred to earlier. Because the TS algorithm always implements the best move available, even if it is uphill, it may take several hundred iterations before the algorithm can recover. Whereas the SA accepts uphill moves with very low probability when it has very nearly converged, the TS will continue to make uphill moves regularly. More fine-tuning of the tabu tenure and diversification rates can probably improve the performance for a specific problem instance, but the SA seems to be more robust in this regard.

Both algorithms did better on the simpler plantation liquidation problems than on the harvest scheduling problems with flow constraints. Fine-tuning of parameters could yield better results but the SA was more robust overall, particularly when a mix of forest types was encountered. Why the TS performed so poorly on these two problem sets but not on the others is unknown.

Execution times remain a concern. The problems used in this study are trivial and yet required significant computational effort to solve. A realistic sized problem would have as many as 100 to 1000 times as many cells as the largest problem used in this study. Faster processors and optimized coding can dramatically improve performance but it may be that long processing times are unavoidable. Given the complexity of the SFPP, significant computational effort is surely worthwhile, but the question is whether the quality of solutions using SA or TS are significantly better than alternative methods such as the Stanley algorithm which yield good results in a short time.

For now, SA seems to be a logical choice for further development of a model formulation for the SFPP. It seems to adapt fairly well to different objective functions and constraints and remains fairly straightforward to implement. Data structures to represent the SFPP need to be carefully developed to maximize the efficiency of the evaluation function, since it is by far the most computationally intensive part of the algorithm and is called many times. For a large-scale instance of the SFPP, efficiency will be an overriding concern to keep execution times reasonably short.

LITERATURE CITED

CLEMENTS, S.E., P.L. DALLAIN, AND M.S. JAMNICK. 1990. An operational, spatially feasible harvest scheduling model. *Canadian Journal of Forest Research* 20 1438–47.

CONNELLY, WILLIAM J. 1988. Area Based Forest Planning. In *The 1988 Symposium on Systems Analysis in Natural Resources*, March 29–April 1, 1988. Asilomar CA. Edited by B. Kent and L.S. Davis., USDA Forest Service Gen. Tech. Rep. RM–161, Rocky Mountain Forest and Range Experiment Station, Fort Collins, CO. pp 131–37.

GLOVER, F. 1990. Tabu Search: A Tutorial. *Interfaces* 20:4, 74–94.

JAMNICK, M.S., AND K.R. WALTERS, 1993. Spatial and temporal allocation of stratum-based harvest schedules. *Canadian Journal of Forest Research* 23: 402–13.

JOHNSON, D.S., C.R. ARAGON, L.A. MCGEOCH, AND C. SCHEVON 1989. Optimization by simulated annealing: an experimental evaluation; Part 1, Graph Partitioning. *Operations Research* 37: 865–92.

JONES, J.G., B.J. MENEGHIN, AND M.W. KIRBY. 1991. Formulating adjacency constraints in linear optimization models for scheduling projects in tactical planning. *Forest Science* 37: 1283–97.

KIRKPATRICK, S., C.D. GELATT, AND M.P. VECCHI, 1983. Optimization by simulated annealing. *Science* 220: 671–80.

LOCKWOOD, C., AND MOORE, T. 1993. Harvest scheduling with spatial constraints: a simulated annealing approach. *Canadian Journal of Forest Research* 23: 468–78.

MENEGHIN, B.J., M.W. KIRBY, AND J.G. JONES. 1988. An Algorithm for Writing Adjacency Constraints Efficiently in Linear Programming Models. In *The 1988 Symposium on Systems Analysis in Natural Resources*, March 29–April 1, 1988. Asilomar CA. Edited by B. Kent and L.S. Davis., U.S. Department of Agriculture, Forest Service. Gen. Tech. Rep. RM-161, Rocky Mountain Forest and Range Experiment Station, Fort Collins, CO. pp 46–53.

MURRAY, A.T., AND CHURCH, R.L. 1995a. Measuring the efficacy of adjacency constraint structure in forest planning models. *Canadian Journal of Forest Research* 25: 1416–24.

MURRAY, A.T., AND CHURCH, R.L. 1995b. Heuristic solution approaches to operational forest planning problems. OR *Spektrum* 17: 193–203.

O'HARA, A.J., FAALAND, B.H., AND BARE, B.B. 1989. Spatially constrained timber harvest scheduling. *Canadian Journal of Forest Research* 19: 715–24.

REMSOFT INC. 1996. *Design and development of a tactical harvest blocking/scheduling tool*. Final Report: Canadian Forest Service, Pacific Region. March 1996. 18p.

SNYDER, S., AND REVELLE, C. 1996. Temporal and spatial harvesting of irregular systems of parcels. *Canadian Journal of Forest Research* 26: 1079–88.

TORRES–ROJO, J., AND BRODIE, J.D. 1990. Adjacency constraints in harvest scheduling: an aggregation heuristic. *Canadian Journal of Forest Research* 20: 978–86.

YOSHIMOTO, A., AND BRODIE, J.D. 1994. Comparative analysis of algorithms to generate adjacency constraints. *Canadian Journal of Forest Research* 24: 1277–88.