Constraint Branching Techniques

Alastair McNaughton
Department of Mathematics
University of Auckland
Private Bag 92019
New Zealand
a.mcnaughton@auckland.ac.nz

David Ryan
Department of Engineering Science
University of Auckland
Private Bag 92019
New Zealand
d.ryan@auckland.ac.nz

Abstract

This talk will outine the applicability of constraint branching particularly in relation to very large problems. It will be discussed in relation to constrained forest harvesting.

Key words: forest harvesting, adjacency branches, area restriction model.

1 Introduction

Many significan applications in OR can be modelled as mixed integer programs. It is common for the shere size of a modern industrial application to preclude most of the usual formulations. In these cases a combination of constraint branching and column generation may be worth considering. The particular application we address is the area-restricted forest harvesting problem.
The basic idea is that we begin with a highly simplified formulation which omits any complcated constraints. This relaxed model is optimized and the relaxed solution is searched for any infeasibilities with respect to the real application. Then, instead of removing these by the inclusion of additional constraints, we impose branches that remove the infeasible solution and permit only the generation of new columns which are feasible with respect to the relevant

restrictions. The success of the method depends on the availability of a very fast column generation technique.
The choice of decision variables should harmonize with the constraint branches. this works best if the decision variables are in a sense orthogonal to the constraint branches. In our forestry application we use road harvest plan variables in which a single variable represents a plan detailing the entire management of all blocks on the road in question throughout the planning horizon. As this is a further comment on an ongoing research project the details of the model formulation are omitted. Please refer to the bibliography for these.

2 The Ideal Constraint Branch

A constraint branch should move the relaxed solution smoothly and swiftly towards the optimal feasible solution. To achieve this a number of things are required. The branch should have a sensible practical meaning. The branch should involve the removal (inclusion) of considerable number of candidate columns. The branch should incorporate time aspects. The branch should have an easily identified most likely side. The selection of branch nodes should allow prioritization such that the relaxed LP solution moves naturally towards those parts of the solution space which have attractive objective values. The branch should help to integerize the problem. Such a branch will likely be particular to the application.

3 A Nuclear Set

We illustrate the concept of a constraint branch with a practical example taken from the area constrained forest harvesting problem. A nuclear set is a local arrangements of harvesting units consisting of a central nucleus surrounded by some perimeter blocks. The *nucleus* is a contiguous block of units with total area less than or equal to the maximum clearfell area. The *perimeter* consists of the surrounding units which are adjacent to the nucleus such that the total area of nucleus plus each individual perimeter unit exceeds the maximum clearfell area. An example of a nuclear set is given in Figure 1. This detail taken from the 400 unit forest is circled in Figure 2. The maximum clearfell area is 30 hectares. Units 75 and 95 form the nucleus, and units 55, 56, 74, 94 and 114 are the perimeter. Unit 96 is not part of the perimeter since its area is too small. A nuclear set is not restricted to the units along 1 road. The example given in Figure 1 spans 4 roads. The concept of the nuclear set lies at the heart of the constraint branch. Figure 2 illustrates the orthogonality of the decision variables and the nuclear sets.

4 The Column Generation

There are many standard approaches to column generation. For a large industrial application involving constraint branching it is imperrative that the column generation be extremely rapid. We use a technique that should be suitable to

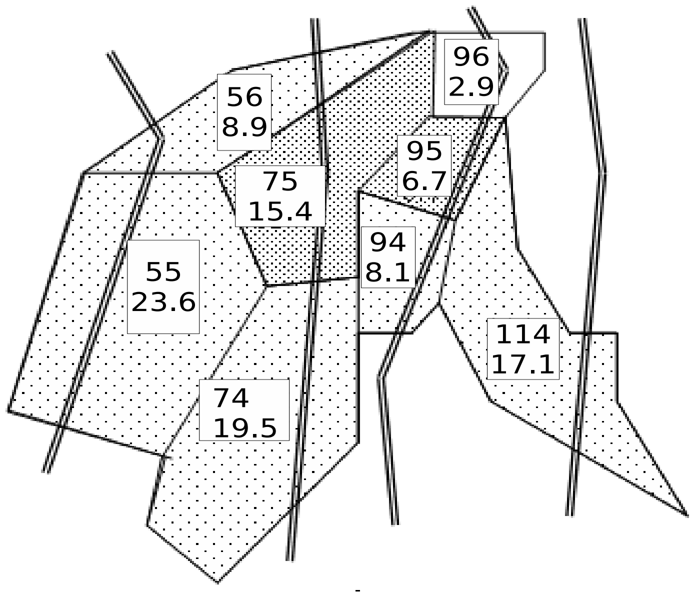

Figure 1: A nuclear set with nucleus (dark shaded) and perimeter (light shaded).

many applications in which the size of the problem stems from combinatorial complexity. We begin with a set of elementary columns each representing the harvest of just 1 unit in 1 particular year.The reduced costs of these columns are then decomposed and recombined so as to construct a new combined column representing the harvesting of several blocks throughout the time horizon so as to achieve optimality with respect to the current branches.

5 The Algorithm

We discuss the solution algorithm for the area restricted forest harvesting problem in the hope that this in some way gives an outline that could be adapted to other applications. The solution algorithm deals with the planning horizon as a single optimization. The strategy is as follows. A list is made of all possible nuclear sets. We then wait to see which of these will be required. For example, the nuclear set in Figure 1 would be included in our list at this stage. However, it would only be used in the solution algorithm if at some stage a relaxed LP solution happened to contain it as part of an adjacency violation.
Figure 3 presents a flow diagram. The forest harvesting problem is initially formulated and solved as a relaxed LP, with no adjacency or integer requirements. A phase of column generation follows with a number of composite road harvest plan variables being added to the model. Optimization (2) follows. The solution obtained is searched so as to find any cases of adjacency violation. This is done by

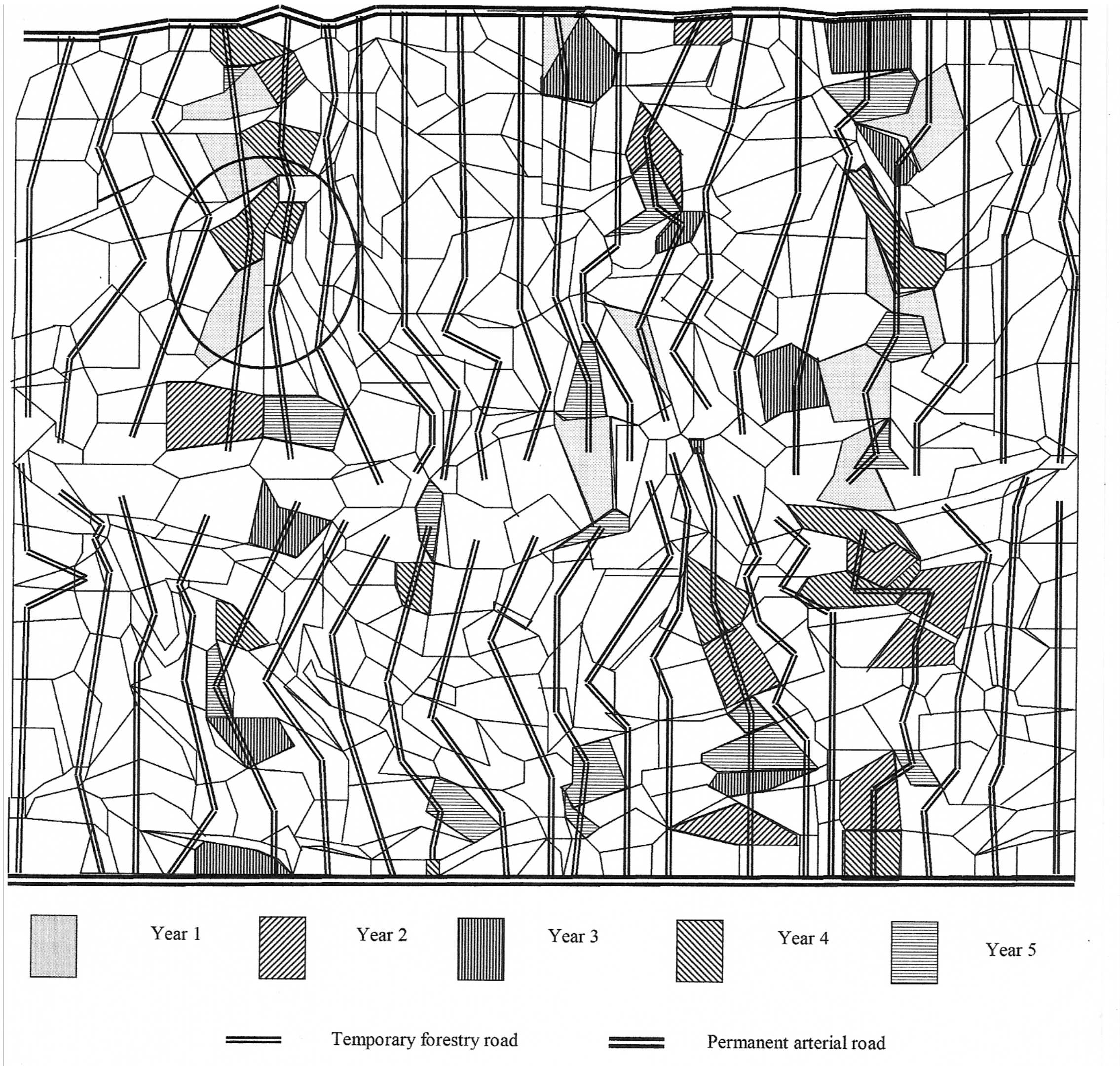

Figure 2: A 400 unit poorly regulated forest. The solution for the trial using a planning horizon of 5 time periods with a green-up of 2 time periods is shown.

a simple scanning process in which each nuclear set on the list is checked sequentially time-wise. An infringement is detected in the form of an identification of a nuclear set with a time interval $[a, b]$, where all the units in the nucleus are harvested within the interval $[a, b]$, with $b - a < T$, and at least 1 unit in the perimeter harvested during the time interval $[b - T + 1, a + T - 1]$. Each iteration only 1 infringement is dealt with.
Each adjacency branch is associated with a nuclear set. In the 1-branch all units in the nucleus are felled within the time interval $[a, b]$. Concurrently with this, all the units in the perimeter are left unharvested during the appropriate time interval, $[b - T + 1, a + T - 1]$. After each adjacency branch has been implemented, the modified linear programme is re-solved, with more column

generation as required. During optimization (1) the new branch is enforced by an artificial penalty on the decision variables which are to be removed by the branch. Column generation then produces several new columns. After this the decision variables which are being removed have their upper bounds set at 0 and optimization (2) follows. If the solution obtained is unacceptable perhaps due to fathoming or to infeasibility, then we back-track. This involves replacing the 1-branch with the 0-branch. The process is repeated until no adjacency violations can be detected. At this stage the solution may still contain fractional values of the decision variables. Integer branches are then used until an integer solution with an acceptable objective value has been obtained. Then the problem is re-optimized with more column generation after every branch. During this integer branching, regular checks are made to detect any further adjacency violations with further adjacency branches are implemented as required. A feasible integer solution is obtained once no adjacency violations and no fractional values of decision variables are left.

6 Adjacency Branches

This section is the main distinctive of this paper. Again we treat it by presenting the area restricted forest harvesting problem as a typical example. After each episode of column generation, each nuclear set on the previously prepared list is scanned. Let us suppose an adjacency violation is found with the nucleus felled during a time interval $[a, b]$. If there are several we select the one with the nucleus of greatest yield. We impose the 1- branch which forces the nucleus units to be harvested within the time interval $[a, b]$.

To implement an adjacency branch we impose an upper bound of 0 on every road harvest plan variable that represents the harvesting of any of the nucleus units *outside* the interval $[a, b]$, including the possibility of a null harvest. Also we require there to be no harvesting of any of the perimeter units during the period $[b - T + 1, a + T - 1]$, where T is the number of time periods in the green-up. To do this we impose an upper bound of 0 on all road harvest plan variables that represent the harvesting of any of the perimeter units *inside* the interval $[b - T + 1, a + T - 1]$.

The adjacency branches also tend to remove fractions from the relaxed LP and so work harmoniously with the integer branches. In the trials the algorithm gave precedence to adjacency branches, followed by integer branches. Nuclear sets corresponding to parts of the forest desirable for harvesting take integer values at an early stage during the algorithm. As a consequence they make adjacency branches which occur early in the branch and bound tree. In this way the branching is prioritized in a good way.

Integer branches remove any remaining fractional values from the variables, x_{jn}, in the relaxed LP, so as to obtain an integer solution. First, the required harvesting time for each unit is obtained from the current relaxed LP solution. For some units the result may be a series of fractions, relative to the time periods, which sum to 1. These are spread over a time interval, say $[a, b]$. We find the smallest fraction associated with either a or b across all the possible units. If the

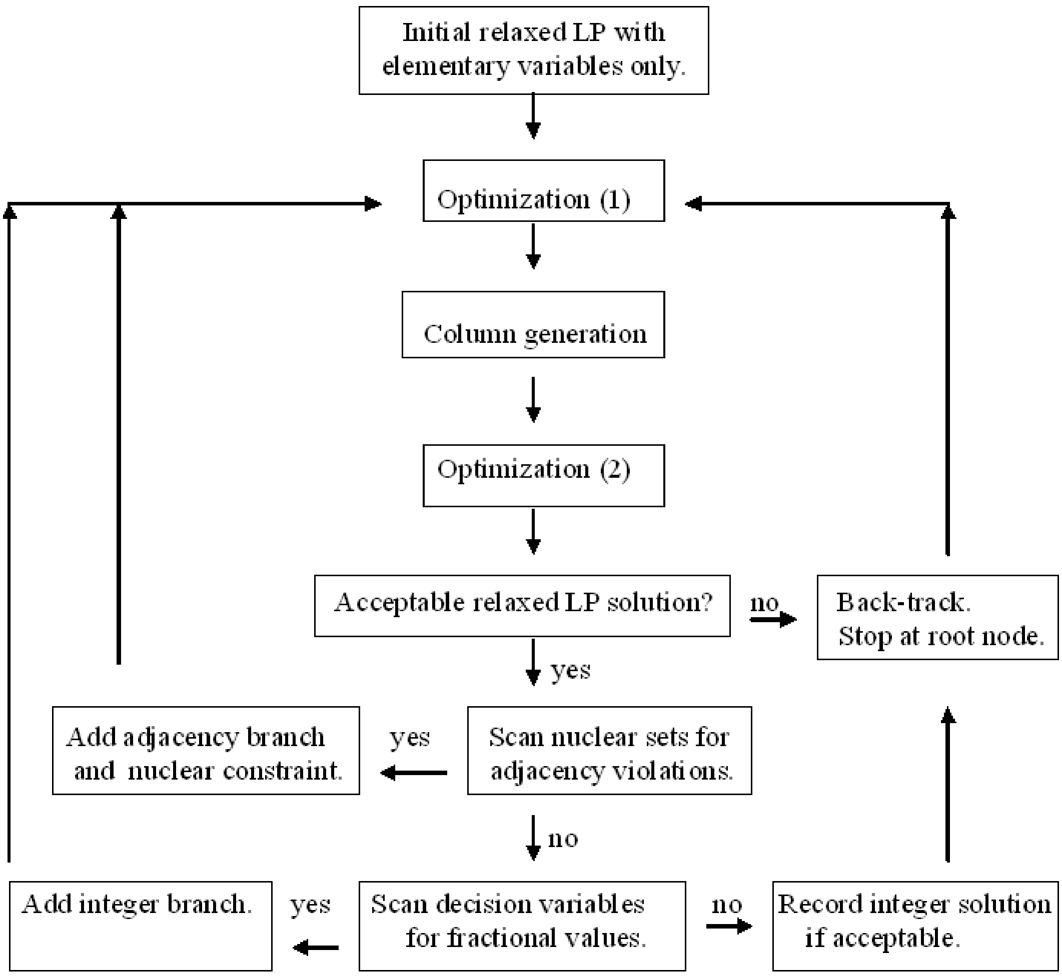

Figure 3: A flow chart of the solution algorithm.

smallest fraction is associated with unit k on road j in time period a, then the 1-branch requires unit k to be harvested after time a. Any variable x_{jn} in which unit k is harvested up to time period a is removed from the problem by having its upper bound set at 0. The associated 0-branch requires unit k to be harvested no later than time period a.

relaxed LP (objective	green-up (time periods)	objective (million $US)	upper bound	optimality gap	time (seconds)	adjacency branches
forest type: poorly regulated						
17.83						
	0	17.77				
	1	17.73	17.81	0.08	34	88
	2	17.62	17.75	0.13	45	142
	3	17.36	17.65	0.29	47	176
	4	17.35	17.51	0.16	47	193
	5	17.02	17.36	0.34	122	216
forest type: well regulated						
15.65						
	0	15.60				
	1	15.50	15.63	0.13	30	96
	2	15.35	15.55	0.20	27	151
	3	15.23	15.45	0.22	57	191
	4	15.18	15.25	0.07	180	240
	5	15.12	15.23	0.11	20	205
forest type: over mature						
20.31						
	0	20.24				
	1	20.20	20.24	0.04	40	60
	2	20.15	20.23	0.08	138	131
	3	20.10	20.12	0.02	228	141
	4	19.96	19.97	0.01	85	204
	5	19.60	19.72	0.12	80	208

Table 1 : Results from trials with a forest of 400 units.

7 Conclusions

Table 1 shows typical performance of the algorithm with respect to the area restricted forest harvesting problem with a variety of data set simulations comprising forests of 400 units. Various forest types have been used so as to test the robustness of the model. The treatment of green-up spanning up to 5 time periods is very significant. In each case the planning horizon is 25 time periods. The level of resolution in the trials indicates that this approach compares favourably with other area restricted forest harvesting models in terms of fine detail, with regard to both time and area. It is hoped the methods presented here may achieve similar improvements in some other applications involving massive combinatorial complexity.

References

[1] Gunn, E.A. and E.W. Richards. 2005. *Solving the adjacency problem with stand-centered constraints*, Can. J. For. Res. 35, pp 832-842.

[2] McDill, M.E., S.A. Rebain and J. Braze. 2002. *Harvest Scheduling with Area-Based Adjacency Constraints*, Forest Science 48, pp 631 - 642.

[3] McNaughton, A.J., M. Rönnqvist and D.M. Ryan. 2000. *A Model which Integrates Strategic and Tactical Aspects of Forest Harvesting*. In System Modelling and Optimization, Methods, Theory and Applications, Edited by M.J.D. Powell and S. Scholtes, Kluwer Academic Publishers Boston, pp 189-208.

[4] McNaughton, A.J., G.D. Page and D.M. Ryan. 2001. *Adjacency Constraints in Forest Harvesting*, proceedings of the ORSNZ, 2001, pp 9-15.

[5] McNaughton, A.J. 2002. *Optimisation of Forest Harvesting Subject to Area Restrictions on Clearfell*, proceedings of the ORSNZ, 2002, pp 307-313.

[6] McNaughton, A.J. 2003. *Adjacency constraints and adjacency branches*, proceedings of the ORSNZ, 2003, pp .

[7] McNaughton, A.J. 2004. *Recent Progress on the Area Restriction Problem of Forest Harvesting*, proceedings of the ORSNZ, 2004, pp .

[8] McNaughton, A.J. and D.M. Ryan. 2007. *Area-restricted forest harvesting with adjacency branches* , proceedings of the ORSNZ, 2007, pp 114-118

[9] McNaughton, A.J. and D.M. Ryan. 2008. *Adjacency branches used to optimize forest harvesting subject to area restrictions on clearfell* , Forest Science 54(4), 2008, pp 442 - 454.

[10] McNaughton, A.J. and D.M. Ryan. 2009. *Area restricted forest harvesting with adjacency branches* , proceedings of the ORSNZ, 2009, pp 110-119

[11] Murray, A. 1999. *Spatial Restrictions in Forest Scheduling*, Forest Science 45(1), pp 45-52.

[12] Murray, A.T. and A. Weintraub. 2002. *Scale and Unit Specification Influences in Harvest Scheduling with Maximum Area Restrictions*, Forest Science 48, pp 779-789.

[13] Vielma, J.P., A.T. Murray, D. Ryan and A. Weintraub. 2003. *Improved Solution Techniques for Multiperiod Area-based Harvest Scheduling Problems*, Systems Analysis in Forest Resources: Proceedings of the 2003 Symposium, pp 285-290.