European Journal of Operational Research 196 (2009) 688–696

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Innovative Applications of O.R.

Multi-commodity supply network planning in the forest supply chain

Satyaveer S. Chauhan [a], Jean-Marc Frayret [b,*], Luc LeBel [c]

[a] Concordia University, Montreal, Quebec, Canada
[b] Ecole polytechnique de Montreal, Quebec, Canada G1K7P4
[c] Faculte de foresterie et de geomatique, Universite Laval, Quebec, Canada

ARTICLE INFO

Article history:
Received 13 April 2005
Accepted 14 March 2008
Available online 26 March 2008

Keywords:
Supply planning
Integer programming
Dynamic programming
Branch-and-price
Cut-to-length timber procurement

ABSTRACT

We consider in this paper a two echelon timber procurement system in which the first echelon consists of multiple harvesting blocks and the second echelon consists of multiple mills (e.g., sawmills), both distributed geographically. Demand is put forward by mills in the form of volumes of logs of specific length and species. Due to the impact of log handling and sorting on cut-to-length harvester and forwarder productivity [Gingras, J.-F., Favreau, J., 2002. Incidence du triage sur la productivité des systèmes par bois tronçonnés. Avantage 3], the harvesting cost per unit volume increases as the number of product variety harvested per block increases. The overall product allocation problem is a large scale mixed integer programming problem with the objective of minimizing combined harvesting and aggregated transportation costs, under demand satisfaction constraints. A heuristic is first introduced then, an algorithm based on the branch-and-price approach is proposed for larger scale problems. Experimentations compare solutions found with the heuristic with the corresponding optimal solutions obtained with both Cplex (using the branch-and-bound approach) and the branch-and-price approach. Results demonstrate the good performance level of the heuristic approach for small scale problems, and of the branch-and-price approach for large scale problems.

1. Introduction

The forest supply chain is very specific and quite different from the traditional manufacturing supply chains for many reasons (Martell et al., 1998; Bredström et al., 2004; Epstein et al., 1999; Rönnqvist, 2003). For instance, forest management and fiber supply planning involves many decision problems ranging from long term strategic forest management (e.g., forest treatment selection, main access road building, forest camp building) to tactical planning (e.g., localization and selection of forest blocks to harvest over a one year horizon, secondary access road building, on site inventory location selection) to short term harvest operations planning (e.g., machines allocation to blocks, bucking pattern rules selection, detailed product allocation to mills). Furthermore, because it is difficult and costly (whenever possible) to have detailed and accurate information about supply availability (particularly regarding tree diameter, taper function, species, diameter and knot internal location), forest planning is done in a highly stochastic context where information quality is rather poor, especially in natural forest. Robust planning under such uncertainty is thus a challenge that must be addressed by foresters, although it is still mainly a subject for researchers. For instance, Boychuk and Martell (1996) address the problem of timber supply under risk of fire. Along the same line, Beaudoin et al. (2007) address plan robustness assessment taking into account multiple sources of uncertainties related mainly to supply availability and machine capacity.

In this paper, we propose to address the short term supply network planning problem which is to decide what timber assortment should be produced in pre-selected blocks in order to fulfill the short terms needs of many geographically distributed mills. The remainder of this paper is organized in five sections. The next section proposes a literature review of the problems addressed in this paper. Then, Sections 3 and 4 introduce respectively the problem and its specific formulation. Section 5 is dedicated to the branch-and-price approach. Empirical evaluation is then presented in Section 6. Finally, a conclusion summarizes the proposed approach and presents some research perspectives.

2. Literature review

Due to the divergent nature of the cut-to-length harvesting process (i.e., trees are broken down into various type of logs), cut-to-length timber procurement planning must address two classic problems. The first problem is a multi-commodity supply planning problem with multiple sources of supply and multiple points of consumption. Then, because one also must decide how trees

* Corresponding author. Tel.: +1 514 848 2424x2978.
E-mail addresses: sschauha@jmsb.concordia.ca (S.S. Chauhan), jean-marc.frayret@cirrelt.ca, Jean-Marc.Frayret@polymtl.ca (J.-M. Frayret), Luc.Lebel@sbf.ulaval.ca (L. LeBel).

0377-2217/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2008.03.024

should be cut into logs of different length, the second problem is similar to the classical cutting-stock problem. If treated sequentially, not all supply plans are feasible because the heterogeneous nature of the forest and trees obviously constraint the supply volume and the type of logs that can actually be harvested. For instance, if the supply planning decision does not take into account the diameter distribution of trees, it is possible that the resulting supply plan proposes to cut long type of log when trees diameter are too small to permit such a mix of products. Consequently, these decisions should be somehow coordinated, if not taken simultaneously. It is typically a problem of integrated process planning and operation planning in a context of divergent transformation process.

Multi-commodity supply network planning problems have been known and studied by many authors. Typically, in the multi-commodity distribution problem, a sub-set of plants is identified from a given set in order to satisfy the demand of a list of customers for one or more commodities. This kind of problems is usually characterized by a set of costs (linear or non-linear), including transportation, production and a fixed cost of choosing a particular plant. The problem becomes complex when there are capacity constraints associated with plants. Krarup and Pruzan (1983) prove that the problem is NP hard when plants have limited capacities. For an in-depth review of multi-commodity supply network planning literature, the interested reader is referred to Aikens (1985).

Although its finality is similar, the problem addressed in this paper is quite different from the classical multi-commodity distribution problem. It is different in the sense that there is no fixed cost for choosing any particular block to harvest. Instead, a variable production cost depends upon the number of different product types to produce in each block (see Fig. 1).

Although we are not addressing the cutting-stock problem (i.e., CSP) in this paper, we provide a selection of references on similar problems. There are several variant of CSP treated by several authors for several different context. Oliver (2002) addresses the one-dimensional cutting-stock problem by proposing linear programming models and approaches based on column generation decomposition. Along the same line, Umetani et al. (2003) develop an approach for minimizing the number of patterns used for producing a specific mix of products. Also, Antonio et al. (1999) proposed two methods based on dynamic programming to address a large spectrum of industrial cutting-stock problems. For the cutting-stock problem involved in forest operations planning, also referred to as the forest-level bucking optimization problem, the reader is particularly referred to Arce et al. (2002) and Laroze (1999).

In the cut-to-length setting, addressed in this paper, trees are cross-cut directly in the forest and timber (i.e. logs of various sizes) is supplied to different mills located sometimes far from the forest. Full length stems are thus cross-cut into logs of different sizes according to the mills demand (i.e., demand-oriented bucking selection problem). In this process, two costs play an important role: (i) the cost of harvesting (which includes bucking, handling, and sorting); and (ii) the cost of transportation of the timber to the mills. Because the introduction of a new product type (i.e., log of a specific length and specie) to separate at the stump in the final felling reduces harvester productivity by 1–4% and forwarder productivity by 3–7% (Brunberg and Arlinger, 2001; Gingras and Favreau, 2002), harvesting cost, in each block, is a function of the product-mix and the volume harvested. Given that blocks and mills are geographically distributed, there is an opportunity to reduce mills procurement cost by synchronizing bucking and transportation decisions to decide at the same time what to produce in each block and what to transport to each mill. In Arce et al. (2002) and Laroze (1999), the objective function of the forest-level bucking optimization problem is to maximize the net profit of using specific bucking patterns on trees of specific stands and classes. In these papers, net profit is the maximum of the sum of the profit per stem using the bucking patterns that satisfy an aggregated demand. In other words, demand is only known per product or market type (e.g., export logs, saw logs, pulp logs) and is not segregated by location. Therefore, because net profit is a function of transportation cost and distance between mills, blocks should affect the selection of bucking patterns whenever demand is known as a mix of specific volumes of products per mill to supply. This justifies the need to solve the supply network planning problem taking transportation and harvesting costs into consideration.

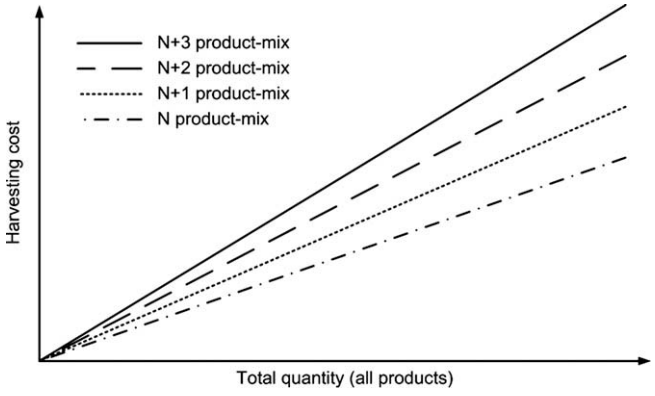

Fig. 1. Harvesting scenarios.

This type of complex problem can be formulated as a mixed integer programming (MIP) problem because the cost of harvesting is a non-continuous linear function of the number of product types to harvest. The difficulty to solve this type of problem is linked to the combinatorial complexity of the problem. The solution space indeed increases exponentially as the problem size increases.

A long body of knowledge deals with a variety of MIP problems, which constitute a subclass of combinatorial optimization problems (Hans, 2001). Many problem specific algorithms exist for finding feasible solutions or even optimal solutions. Three basic methods, branch-and-bound (Mitten, 1970), Cutting plane algorithm, and Dynamic programming are widely used for solving such problems (Winston, 1993; Murty, 1988; Nemhauser and Wolsey, 1988). Sometimes these methods are used in conjunction with others. For example, when cutting plane method is used with branch-and-bound, the technique is known as branch-and-cut algorithms (Hoffman and Padberg, 1985), or when column generation (Dantzig and Wolfe, 1960) is used in conjunction with branch-and-bound, it is referred to as branch-and-price (Vance et al., 1994). The algorithms using branch-and-bound require good starting bound which is usually obtained by LP relaxations. Lagrangian relaxation (Geoffrion, 1974; Fisher, 1985) is one of the popular approaches to obtain such a good bound.

Solving combinatorial optimization problems requires a trade-off between computational time and quality of the solution. In this paper we present a heuristic approach in order to quickly find a good solution, which can be used as an upper bound for the branch-and-bound algorithm developed to reach the optimal solution.

3. General problem introduction

Because of the inherent complexity of the problem introduced in the previous section (i.e., a combined problem of multi-commodity supply network planning and forest-level bucking optimization), we propose to decompose it into two inter-related problems to be solved iteratively until a solution is reached. The

theoretical framework presented in Schneeweiss (2003) is used here to model the proposed planning process (see Fig. 2). In brief, Schneeweiss models a complex planning problem by decomposing it into a top (i.e., an aggregated decision problem such as the capacity planning of multiple facilities or a complex production system) and a base level (i.e., a more detailed decision planning such as the operations planning for a short time horizon) problems. Because the base level problem constraints and/or contribute to the objective of the top level problem, the top level must be able to adequately anticipate the influence the base level might have on the top level objective if a particular decision (i.e., top level instruction) is taken. To do so, the top level possesses some form of anticipation function of the base level. Schneeweiss (2003) identifies four generic types of such functions: perfect; approximate explicit reactive; implicit reactive; and non-reactive.

More specifically, in the context of our problem, the top level represents an extended version of the classic multi-commodity supply network planning problem that takes into account the product-mix dependent production cost. Because taking into account at this planning level all the details about tree diameter distribution and taper function would result in an even larger scale problem to solve, these parameters are only modeled as aggregated available fiber volume per species and block. Furthermore, at this level no restriction regarding the product-mix to be harvested is considered. In other words, we consider that the selection of particular bucking pattern rules (defined here as a set of bucking patterns specified for each particular class of tree diameter) does not constraint the mix of product that can be produced. Indeed, many sets of such rules can generate the same product mix, although the resulting resource utilization efficiency may differ from one set of rules to the other. From a modeling perspective, this form of anticipation of the base level is non-reactive as it is not influenced by the instruction of the top level. In Schneeweiss (2003) terminology, this type of anticipation is called non-reactive because the top level is not explicitly aware of the objective function of the base level. Furthermore, only general features of the base level, such as the aggregated supply availabilities, are considered. This approach does not guaranty that the instructions of the top level (i.e., a tentative supply planning solutions of the top level) will be feasible (i.e., there is at least one set of bucking pattern rules that satisfies the top level instruction). That is why the base level aims at finding for each block independently the best set of bucking pattern rules, also called the reaction of the base level, that satisfies the production plan (i.e., target volumes of certain products type for each block) of the top level taking into account detailed information about tree diameters distribution and taper function while minimizing resource utilization. The base level consequently consists in many independent bucking decision sub-systems. Due to the non-reactive nature of the anticipation, the instruction of the top level may not be always feasible because of an overestimation of resource availability. This is why, once the base level reaction is completely computed for each block, the planning process consist of analyzing at this stage whether the overall supply plan remains feasible or not. If not, the aggregated availability must be adjusted for each block (i.e., increased if availability remains, or decreased in case of local un feasibility) in order to best match the base level reaction. Then, the top level can compute another supply plan that takes into account the new resource availability constraints. This iterative process continues until a feasible solution is found or mills demand must be adjusted.

Within this overall iterative planning process, this paper focuses on the top level normative decision problem. More specifically, the problem we propose to address is a single-period multi-commodity supply planning problem with multiple sources of supply (i.e., blocks to be harvested) and multiple points of consumption (i.e., mills) and with product-mix dependent production cost. In this particular setting, blocks are completely harvested (i.e. no partial harvesting) during the planning period considered. The selection of the blocks to be harvested during this period to satisfy mills demand is part of another decision module that is not

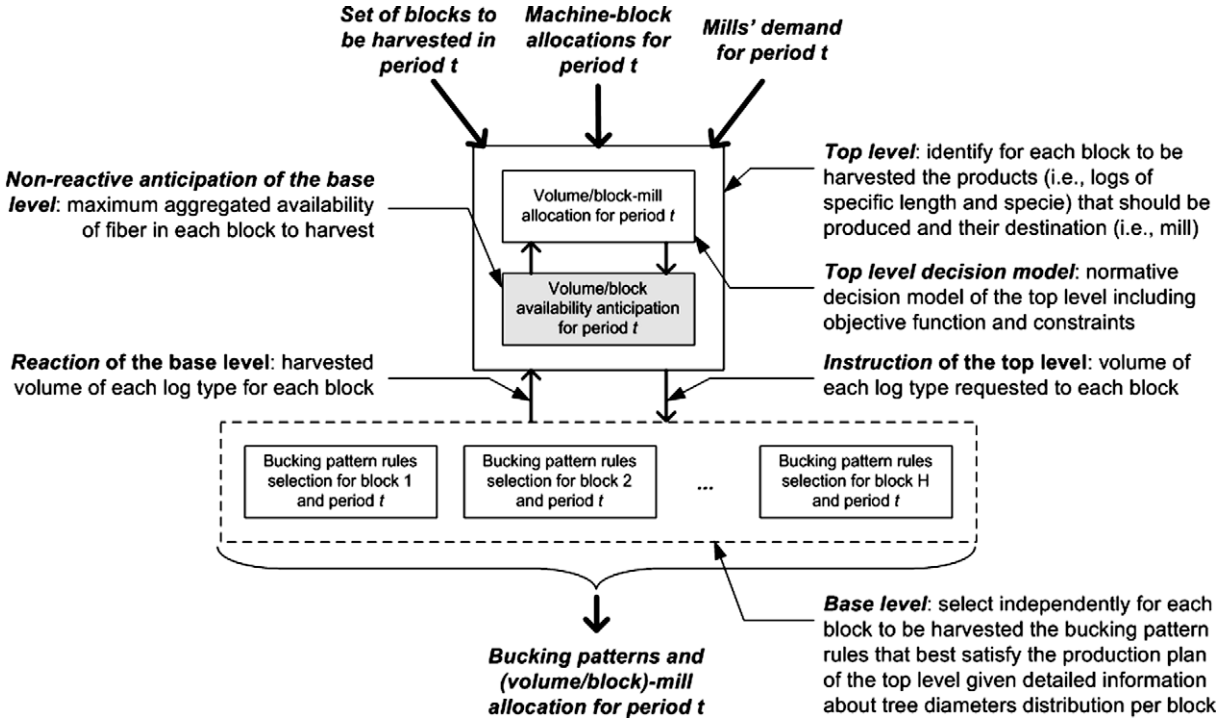

Fig. 2. Supply planning.

described in this paper. The interested reader is referred to Beaudoin et al. (2007) for more information.

Concerning demand, we consider that the supply needs of the mills are known and specified in terms of volume (i.e., m^3) of logs of different sizes in length, minimum small end diameter (SED) and species. As mentioned earlier, we assume that all blocks are capable of providing all length of products in limited amounts. In other words, the aggregated available volume for each species in each block is known and adjusted iteratively if necessary. In a context where detailed information about tree diameter distribution is not known in detail, using aggregated information is the easiest way to anticipate what could be harvested from specific blocks. The obvious limit of this approach is that the minimum SED allowed by mills for each product type makes it more or less difficult to realize every supply plan (i.e., find a set of bucking pattern rules). For instance, producing a single product type in a block, while having a rather large allowed minimum SED for that product type (which is typical for long length log type), generally results in an artificially reduced resource availability (because the part of the tree above the minimum SED is not available to produce logs). That is why, the introduction of another product type in that block with a smaller minimum allowed SED may increase resource availability (and production cost as well). A trade-off between demand satisfaction and production cost is then necessary. Furthermore, although aggregated information about resource availability is often the only source of information available to foresters, more and more companies show interest in having more detailed information in order to better adjust their harvesting operations to their rapidly changing needs.

4. Specific problem formulation

The objective of the problem we address in this paper is to decide the mix and volume of product that should be harvested in each block so as to satisfy the demand of each mill at the minimum combined cost of harvesting and transportation to the mills. As mentioned earlier, the unit harvesting cost per block increases with the number of product types to be produced in the block. Consequently, we assume in this model that harvesting cost depends upon the number of product types produced in a block, as well as the volume of production for each product type. Next, transportation cost is linear with respect to the volume to be transported and depends upon the distance between blocks and mills. Given these specifications, the MIP formulation of this problem follows:

H	set of harvesting blocks
S	set of mills
L	set of log/product types
K	set of species
h, s, l, k	indexes of blocks, mills, logs, and tree species, respectively
N	$\|L\|$ (maximum number of product types)
$x^h_{k,l,s,n}$	number of logs produced at h, of log l, using species k for mill s (decision variable) if n product types are produced in that block
$a^h_{l,s}$	unit transportation cost between h and s for log type l (dollar/m^3)
V^h_k	maximum volume of species k available at h (m^3)
I^h_n	binary indicator: takes value 1 if h produces n product types; otherwise 0
$J^h_{k,l}$	binary indicator: takes value 1 if block h produces log type l of species k; otherwise 0
$\alpha_{k,l}$	volume of a log type l of species k (m^3/log)
$b_{h,n}$	unit harvesting cost if n product types are produced at harvesting block h (dollar/m^3)
$d_{l,k,s}$	demand of sawmill s for log type l of species k (m^3)

P_1 Min transportation and production cost

$$\sum_{n=1}^{N}\sum_{h\in H}\sum_{k\in K}\sum_{l\in L}\sum_{s\in S}(b_{h,n}+a^h_{l,s})\cdot\alpha_{k,l}\cdot x^h_{k,l,s,n}$$

Subject to

$$\sum_{n=1}^{N}\sum_{h\in H}\alpha_{k,l}\cdot x^h_{k,l,s,n}=d_{l,k,s}\quad\forall l\in L\quad\text{and}\quad s\in S,k\in K. \tag{1}$$

$$\sum_{k\in K}\sum_{l\in L}\sum_{s\in S}x^h_{k,l,s,n}\leqslant Z\cdot I^h_n\quad\forall h\in H\quad\text{and}\quad n\in\{1,2,\ldots,N\}. \tag{2}$$

$$x^h_{k,l,s,n}\leqslant Z\cdot J^h_{k,l}\quad\forall h\in H,\quad k\in K,\quad l\in L,\quad s\in S,\quad n\in\{1,2,\ldots,N\} \tag{3}$$

$$J^h_{k,l}=\{0,1\}, \tag{4}$$

$$\sum_{k\in K}\sum_{l\in L}J^h_{k,l}=\sum_{n\in 1,2,\ldots N}n\cdot I^h_n\quad\forall h\in H, \tag{5}$$

$$\sum_{n\in 1,2,\ldots N}I^h_n\leqslant 1\quad\forall h\in H, \tag{6}$$

$$I^h_n=\{0,1\}\quad\forall h\in H\quad\text{and}\quad n\in\{1,2,\ldots,N\}, \tag{7}$$

$$\sum_{l\in L}\sum_{s\in S}\sum_{n\in\{1,2,\ldots,N\}}\alpha_{k,l}.x^h_{k,l,s,n}\leqslant V^h_k\quad\forall k\in K,\quad h\in H. \tag{8}$$

$$x^h_{k,l,s,n}\geqslant 0\quad\forall h\in H,\quad k\in K,\quad s\in S\quad\text{and}\quad n\in\{1,2,\ldots,N\}. \tag{9}$$

Constraint (1) stipulates that the total quantity, corresponding to a particular product type, received by any mill from all harvesting blocks must match the corresponding demand of that product type. Constraint (2) guarantees that the decision variables corresponding to n-product type takes positive values only if the corresponding binary variable I^h_n is positive. Z is a big number, bigger than the combined demand. Constraint (3) and (4) state that if block h is involved in the harvesting of product of type l and species k, then the corresponding binary variable $J^h_{k,l}$ should take a positive value. Constraints (5) to (7) are used to select the appropriate cost parameters in the objective function. Finally, constraint (8) corresponds to the resource constraints which indicate that total harvested volume should not go beyond the aggregated resource availability.

Problem P_1 is a large MIP problem. The main challenge to solve it arises from the dependence of one binary variable over others. In other words, the setting of one binary variable to one (selection of a particular production scenario) forces all other inter-related binary variables to become zero, which in turn invalidates the current set of cost in the objective function, and forces the system to optimize with the new set of parameters. As stated previously, branch-and-bound approach is required to reach the optimal solution. In the approach proposed here, we first present an algorithm to find a good and feasible solution that can be used as an upper bound for the branch-and-bound algorithm if the optimal solution is of prime importance.

5. Properties and algorithm

In this section we propose to analyze some of the properties of a local optimal solution, and develop algorithms to solve the problem.

5.1. Optimal algorithm for a special case

Let us consider a case of single sawmill and multiple harvesting blocks such that the transportation cost is either negligible compare to harvesting costs or the same for all blocks. Let us also assume that

the available volume of fiber in each block is sufficient to meet the mills' total demand. In the above setting we can solve the problem to optimality using the algorithm described below. A property of such model is that the cost structure is only influenced by the unit harvesting cost of each block (itself being a function of the number of product type harvested) and not by the total volume harvested in the corresponding blocks. Indeed, because of the sufficient resource availability in each block, splitting the mills demand for a particular product type would only results in increasing the unit production cost of at least one block. Furthermore, this unit cost is only specified for a given product type number in terms of volume (m^3) of harvested fiber and not for a particular product-mix. Consequently the objective becomes to decide how many product types to produce in each block. This special case of the problem can be solved using an adaptation of an algorithm, based on dynamic programming and presented in Chauhan et al. (2002).

Let us define a function $\phi(h,a)$ where h is the block index and a denotes the number of product type harvested in this block. $\phi(h,a)$ denotes the minimum cost of harvesting a product types in blocks 1 to h. For example $\phi(3,8)$ denotes the cost of harvesting eight product types from the first three (indexed 1, 2 and 3) blocks. We also define $X[h,a]$ a two-dimension array which denotes the number of product types harvested in block h if a total of a product types are to be harvested in the first h harvesting blocks (i.e., from 1 to h).

The proposed recursive algorithm follows:

Algorithm 1

1. Set $\phi(h,0)=0$ for $h=1,2\ldots,H$.
2. Set $\phi(1,a)=a.b_{1,a}$ and $X[1,a]=a$ for $a=1,2\ldots,A$.
3. For $h=2,\ldots,H$
   For $a=1,2,\ldots,A$
   i. Set $U=\mathrm{Min}_{1\leqslant x\leqslant a}\{\phi(h-1,a-x)+x\cdot b_{h,x}\}$.
   ii. If $\phi(h-1,a)<U$ then set $\phi(h,a)=\phi(h-1,a)$ and $X[h,a]=0$. else
   iii. Set $\phi(h,a)=U$ and $X[h,a]=x$
   In this case $X[h,a]$ contains the x for which U is found
4. Stop.

5.1.1. Explanation

Let us consider that we are dealing with the step where $h=3$. At this step, the values of all $\phi(2,a)$ for $a=1,2,\ldots,A$ have already been calculated. Now at the step (i) of the algorithm, the cost of harvesting $a=1,2,\ldots,A$ products from the first three blocks is computed. Assume $a=5$, we want to know how much it costs to produce five product types using the first three harvesting blocks (since $h=3$). The available options are to produce either one product type in the third block and four product types in the first two blocks, or two product types in the third block and three product types in the first two, and so on. Since the harvesting cost of all combinations of producing 1, 2,...,5 product types using the first two blocks is already calculated, the cost of producing five product types using the first three blocks is easy to compute. At this step, it is possible that harvesting all five product types from the first two blocks is cheaper than sharing the product types among three blocks. If it is the case, at step (ii), $X[3,5]$ is set to 0. In other words, it means that the third block produces nothing if the objective was to harvest five product types using the first three blocks. At step (iii), the cost is set in $X[h,a]$ if the third block produces at least one product type.

5.1.2. Example

In order to help the reader better understand this procedure, let us consider the following example. Assume there are three harvesting blocks and three product types, and the harvesting costs are as follows: $b_{1,1}=0.5$, $b_{1,2}=0.6$, $b_{1,3}=0.7$, $b_{2,1}=0.56$, $b_{2,2}=0.6$, $b_{2,3}=0.71$, $b_{3,1}=0.6$, $b_{3,2}=0.62$, $b_{3,3}=0.65$.

Step 1:
$\phi(1,0)=0,\phi(2,0)=0\phi(3,0)=0.$
Step 2:
$\phi(1,1)=0.5,\quad \phi(1,2)=1.2,\quad \phi(1,3)=2.1.$
$X[1,1]=1,\quad X[1,2]=2,\quad X[1,3]=3$
Step 3:
(h = 2, a = 1), $\phi(2,1)=0.5,\quad X[2,1]=0$
(h = 2, a = 2), $\phi(2,2)=1.06,\quad X[2,2]=1$
(h = 2, a = 3), $\phi(2,3)=1.7,\quad X[2,3]=2$
(h = 3, a = 1), $\phi(3,1)=0.5,\quad X[3,1]=0$
(h = 3, a = 2), $\phi(3,2)=1.06,\quad X[3,2]=0$
(h = 3, a = 3), $\phi(3,3)=1.66,\quad X[3,3]=1$

In this example we want to know how many product types to produce in each block. Since our objective is to harvest three product types using three blocks, $X[3,3]$ shows that it is optimal to harvest only one product in block three. Then, there are two products remaining and two blocks. $X[2,2]$ shows that it is optimal to harvest only one product in block 2. Once again, there is 1 product remaining and 1 block. $X[1,1]$ proposes to harvest only one product in block 1.

Now, assume that we want to harvest three products using the first two blocks. Then $X[2,3]$, proposes to harvest two products in block 2 and 1 product in block 1.

The computation of each $\phi(p,a)$ requires $A+1$ steps. Consequently, the total steps involve in the algorithm are $\sum_{k=0}^{A}\sum_{h=1}^{H}(A+1)$ or $A\cdot H\cdot(A+1)$ to find the optimal value of the special case problem.

The above algorithm identifies how many product types should be harvested from each block. The next step is to identify specific product types and corresponding volume to be harvested in these blocks. This step is rather straight forward. To do so, we first sort the blocks selected for harvesting in increasing order of harvesting cost (note that the above algorithm will tell how many product types to be harvested from each block and therefore the harvesting cost for the block). Next we sort the product types in decreasing order of demanded volume. Finally, we select the first block on the list and allocate as many number of product types from the product type list as it is selected for that block. Then, we take the second block and do the same until no blocks are left in the block list.

5.2. General case

For the study of the general case we now drop the index n, which correspond to the cost type, from $x^h_{k,l,s,n}$. Now the following holds for the general case.

Proposition 1. *There exists an optimal solution in which, for any mill s and for a given product type l of a specific species k*

$$x^h_{k,l,s}\in\{0\}\cup\{d_{l,k,s}\}\quad \textit{for}\quad h\in E(s), \tag{10}$$

where $E(s)\subset\{H\}$ or

$$x^h_{k,l,s}\notin\{0\}\cup\{d_{l,k,s}\}\quad \textit{for}\quad h\in\{1,2,\ldots,H\}\setminus E(s). \tag{11}$$

But in this case $V^h_k-\sum_{s\in S}\sum_{l\in L}\alpha_{k,l}\cdot x^h_{k,l,s}<\alpha_{k,l}\cdot x^h_{k,l,s}$ for $h\in H\setminus E(s)$ except may be for at most one h.

Proof. Let $X=\{x^h_{k,l,s}\}$, $h\in H$, $s\in S$ be a feasible solution to the problem. Assume that the supply corresponding to block $h_1,h_2\in H$, respectively $x^{h_1}_{k,l,s}$ and $x^{h_2}_{k,l,s}$, do not verify the conditions of Proposition 1 for sawmill s. In other words

$x^{h_1}_{k,l,s} \notin \{0\} \cup \{d_{l,k,s}\}$ and $V^{h_1}_k - \sum_{s\in S}\sum_{l\in L} \alpha_{k,l}.x^{h_1}_{k,l,s} \geqslant \alpha_{k,l}$

and

$x^{h_2}_{k,l,s} \notin \{0\} \cup \{d_{l,k,s}\}$ and $V^{h_2}_k - \sum_{s\in S}\sum_{l\in L} \alpha_{k,l}.x^{h_2}_{k,l,s} \geqslant \alpha_{k,l}$

Assume that the cost per m^3 of harvesting and transporting from h_1 is higher than the cost from h_2, we have

$$(b_{h_1,n} + a^{h_1}_{l,s}) \geqslant (b_{h_2,n} + a^{h_2}_{l,s}). \quad (12)$$

Now choose $\delta = \text{Min}(V^{h_2}_k - \sum_{s\in S}\sum_{l\in L}\alpha_{k,l} \cdot x^{h_2}_{k,l,s}, x^{h_1}_{k,l,s})$ and set $y^{h_2}_{k,l,s} = x^{h_2}_{k,l,s} + \delta$, $y^{h_1}_{k,l,s} = x^{h_1}_{k,l,s} - \delta$. Now define the new solution set $Y = \{y^h_{k,l,s}\}$, by replacing in X, $x^{h_1}_{k,l,s}$ by $y^{h_1}_{k,l,s}$ and $x^{h_2}_{k,l,s}$ by $y^{h_2}_{k,l,s}$. The new solution obtained (i) remains feasible because the harvested volume is only switched from one block to the other with respect to availability constraints; (ii) the harvesting cost function corresponding to block h_2 remains the same as before since the above mechanism do not add any new product type in forest h_2 (i.e., only a volume of a product type and species already harvested is added); (iii) the number of product types harvested in block h_1 as well as the total harvesting cost of the block may reduce by one unit if $\delta = x^{h_1}_{k,l,s}$.

Consequently, from the inequality (12) and the fact that costs are positive $(b_{h_1,n} + a^{h_1}_{l,s}) \cdot x^{h_1}_{k,l,s} + (b_{h_2,n} + a^{h_2}_{l,s}) \cdot x^{h_2}_{k,l,s} \geqslant (b_{h_1,n} + a^{h_1}_{l,s}) \cdot y^{h_1}_{k,l,s} + (b_{h_2,n} + a^{h_2}_{l,s}) \cdot y^{h_2}_{k,l,s}$. The above inequality shows that any solution not verifying the conditions of Proposition 1 cannot be the optimal solution because there is potentially at least one better solution. This completes the proof. □

Now, if two or more sawmills (for a particular product) are supplied from two or more blocks, then the satisfaction of Proposition 1 does not guaranty that the solution is optimal, especially if the condition of the Proposition 2 is satisfied.

Proposition 2. *Assume that the conditions of Proposition 1 apply, then if* $x^p_{k,l,q} \notin \{0, d_{l,k,q}\}$ *with* $p \in \{h_1, h_2\} \subseteq H$ *and* $q \in \{s_1, s_2\} \subseteq S$, *then the solution is not optimal.*

Proof. Let X be a set of feasible solutions which does not satisfy the conditions of Proposition 2. For any pair of harvesting blocks h_1, h_2 and sawmills s_1, s_2, either of the following must hold

$$\left(b_{h_1,n_{h_1}} + a^{h_1}_{l,s_1}\right) + \left(b_{h_2,n_{h_2}} + a^{h_2}_{l,s_2}\right) < \left(b_{h_1,n_{h_1}} + a^{h_1}_{l,s_2}\right) + \left(b_{h_2,n_{h_2}} + a^{h_2}_{l,s_1}\right), \quad (13)$$

or,

$$\left(b_{h_1,n_{h_1}} + a^{h_1}_{l,s_1}\right) + \left(b_{h_2,n_{h_2}} + a^{h_2}_{l,s_2}\right) \geqslant \left(b_{h_1,n_{h_1}} + a^{h_1}_{l,s_2}\right) + \left(b_{h_2,n_{h_2}} + a^{h_2}_{l,s_1}\right). \quad (14)$$

Now, define $\delta_1 = \text{Min}(x^{h_1}_{k,l,s_2}, x^{h_2}_{k,l,s_1})$ and $\delta_2 = \text{Min}(x^{h_1}_{k,l,s_1}, x^{h_2}_{k,l,s_2})$. If (13) holds (the opposite is true if (14) holds) then define a new solution set Y same as set X and set the following variables as follows: $y^{h_1}_{l,k,s_1} = x^{h_1}_{l,k,s_1} + \delta_1$, $y^{h_2}_{l,k,s_2} = x^{h_2}_{l,k,s_2} + \delta_1$, $y^{h_1}_{l,k,s_2} = x^{h_1}_{l,k,s_2} - \delta_1$ and $y^{h_2}_{l,k,s_1} = x^{h_2}_{l,k,s_1} - \delta_1$.

The new solution Y remains feasible because the harvested volume is only switched from one block to the other with respect to availability constraints. Also, the above procedure does not add any new product type to any block and may even reduce the number of product types from a block. In the resulting solution, the harvesting cost function associated with the blocks either reduces or remains the same. Even if the harvesting cost remains the same for all blocks, the total cost corresponding to the new solution reduces according to relation (13). □

5.3. Scenario improvement heuristic

The Propositions 1 and 2 provide necessary conditions for a solution to be optimal. In this section we develop a heuristic approach which always satisfies the above conditions and gradually improve the solution. The idea is to start with a given scenario (i.e. chosen product-mix for each block) and improve it (i.e. change the product-mix for one or more block) in order to reduce the overall cost. Once harvesting scenario is set the problem becomes a simple linear programming problem and because of the structure we can use well-known transportation simplex algorithm (by balancing the demand and supply) to find an optimal solution for the given harvesting scenario. The steps of the algorithm are as follows:

1. Introduce the harvesting cost of each block associated with the current solution (based on current product-mix).
2. Compute the reduced cost corresponding to all non-basic variables for the current solution. Select the M variables with highest reduced cost. The value of M depends upon the user. For a given set of costs, the problem is a linear programming problem from which it is easy to calculate dual cost vector and reduced costs.
3. For each M decision variable, using Proposition 1 and 2, we check if the cost can improve by tentatively introducing the variables into the basis. Then we calculate the real cost (based on the real allocation) for each M cases.
4. We select the variable which improves the cost the most and introduce it in the basis. If this solution is better than the best solution we obtained so far, we replace the best solution by the current solution. If the cost does not improve for all the selected M non-basic variables, then we stop, otherwise go to step 1.

In the algorithm we check M number of non-basic variables because the cost does not remain the same when we change the number of product type to be harvested (i.e., as the allocation changes, the harvesting cost may also change). If the cost were constant, then selecting a variable with the positive reduced cost would definitely improve the solution (simplex method). We choose M variables to investigate based on the experience from the experimentation.

The above approach is good to get a good starting upper bound for the problem. Since, the algorithm does not involve binary variables, the total number of variables required by the algorithm are $H * (S * K * P + K)$ whereas the number of ordinary variables required in the BIP formulation is $K * P * (H * S * K * P)$ and the number of binary variables are $H * (2 * P * K)$. The second advantage of the algorithm is that the solution is always feasible whereas the solution obtained by solving the relaxed problem of P_1, to get an initial lower bound, may not be feasible.

For real instances of the problem, the algorithm performed very well because the harvesting and transportation costs generally differ significantly from one forest to other. If these costs are very close to each other, then the algorithm performance tends to deteriorate. This motivated us to develop another approach which is capable of providing better results. In the next section we propose branch-and-price approach for the exact solution. As mentioned before, this level of the problem has missing information about the type of patterns used in the forest (which depends upon trees distribution in the forest, taper profile, the demand assigned to the block, etc.). In this approach we assume that a particular product must be supplied to a sawmill by a single block. In other words we assume that the capacity of the block is sufficient. We will use the above heuristic to compute the initial starting solution and the upper bound.

6. Branch-and-price approach

In this section we first propose a different formulation of the above problem and then discuss the solution approach. We also

use additional notations to simplify the formulation. Let r denotes the total number of product types demanded by all sawmills, i.e., $r = |\{1 \mid d_{l,k,s} \geqslant 0, l \in L, k \in K, s \in S\}|$ where $|\cdot|$ represent the cardinality of a set. We define the binary vector $[B]_{r\times 1}$ and its components $b_{k,l,s}$, where these components sequentially represent the positive value if demand of product type (k,l) for sawmill s is positive else zero. In other words, each component of B represent a particular item for a particular sawmill. For instance, if we have two log lengths, two species and three sawmill then the first component of B correspond to $d_{1,1,1}$, the 2nd correspond to $d_{1,2,1}$, the 3rd correspond to $d_{2,1,1}$, the 5th correspond to $d_{1,1,2}$ and so on.

We similarly define a column vector $[X^h]_{r\times 1}$, which represents the supply for sawmill s from block h. Consequently, each component of X represents the supply of a particular item for a given sawmill. For example, the 5th component of $[X^h]$ will represent supply of wood type $(1,1)$ (log type 1, species type 1) to sawmill 2 by block h. The possibility of having several feasible harvesting plans for each block resulted in several columns (i.e., X^h) for the corresponding block h. We denote by G^h the set of all feasible columns corresponding to block h. In order to identify each column of G^h, we introduce the index i (i.e., X_i^h is ith element of G^h). We denote the components of vector $[X_i^h]$ by $x_{h,i,j}$ where $j \in \{1,2,\ldots,r\}$.

Let us define a binary variables z_i^h, corresponding to each column i associated with block h. z_i^h takes a positive value ($z_i^h = 1$) if the column, i, is selected or zero otherwise. The cost associated with column i of block h is denoted by c_i^h and is calculated on the production and supply represented by the column. Since our objective is to select at most one column for each block, we introduce a convexity constraint for each block. Let Y_i^h be $|H|$-elements column vector of 0s except the hth element which is 1. The jth element of Y_i^h is denoted by $y_{h,i,j}$. Consider now the ith column corresponding to block h as being composed of X and Y and can be denoted by $[X_i^h Y_i^h]$. Using the above notations we can formulate the problem as follows:

$$P_2 \ \text{Min} \sum_{h\in H}\sum_{i\in G^h} c_i^h \cdot z_i^h$$

Subject to

$$\sum_{h\in H}\sum_{i\in G^h} x_{h,i,j}\cdot z_i^h = b_j \quad j \in \{1,2,\ldots,r\}, \tag{15}$$

$$\sum_{i\in G^h} y_{h,i,j}\cdot z_i^h = 1 \quad j \in \{1,2,\ldots,H\}, \quad h \in H, \tag{16}$$

$$z_i^h \in \{0,1\}. \tag{17}$$

The objective is to minimize the combined harvesting and transportation costs. Constraints (15) represent demand satisfaction from combination of sources and (16) force the system to select only a column (harvesting and supply scenario) for each block.

The above formulation guarantees the optimal solution only if set G^h contains all the feasible columns corresponding to block h. Generating all the possible columns beforehand for each block may not be feasible. Therefore, only a limited version of the problem containing few columns is solved, while new profitable columns are iteratively introduced until the optimal solution is found. Since problem P_2 is a binary integer programming model, to solve it, we first optimize its relaxed version (relaxing constrains (16)) and then we use the branch-and-bound strategy to search for the integer solution. We denote by $P_2(RM)$ the restricted and continuous version of the problem P_2.

Let $u_\eta, \eta = 1,2,\ldots,r+|H|$, be the dual variables associated with the current optimal solution of $P_2(RM)$. In order to generate a new column, t, we must solve the following pricing problem:

$$Pr(h)\ \text{Max} \sum_{\eta=1}^{r} u_\eta \cdot x_{h,t,\eta} + u_{r+h}\cdot y_{h,t,h} - c^h$$

Subject to

$$y_{h,t,h} = 1 \tag{18}$$

$$\sum_{s\in S}\sum_{l\in L} d_{l,k,s}\cdot x_{h,t,f(l,k,s)} \leqslant V_k^h \quad \forall k \in K \tag{19}$$

$$x_{h,t,f(l,k,s)} \in \{0,1\} \quad \forall k \in K, \quad l \in L, \quad s \in S, \tag{20}$$

where $f(l,k,s)$ generate the index according to the following formula. $f(l,k,s) = (s-1)\cdot L\cdot K + (k-1)\cdot L + l$. The cost c^h depends upon the number of products harvested in h (see Appendix for the constraints and objective function terms related to c^h). At each iteration we solve $Pr(h)$ for each block and we select the column corresponding to the highest positive reduced cost. We introduce the column in G^h and solve again the $P_2(RM)$. We terminate the column generation algorithm if there is no column with positive reduced cost. The optimal solution of $P_2(RM)$ may not have all z_i^h as integer. We use the branch-and-bound approach at this stage to search for an integer solution. In our approach we use depth first search strategy from left to right.

The idea of introducing z_i^h is to constraint the system to have only single column as additional columns may not represent the same harvesting scenario (i.e. each column represent a harvesting scenario with specific harvesting cost for the entire block). Using classical branching strategy known as variable branching (i.e. selecting a column for the left branch-and dropping the same column for right branch) may create an unbalanced branching tree. In our approach, we use constraint branching. In other words instead of fixing ($z_i^h = 0$) or ($z_i^h = 1$), we fix $x_{h,i}$. In simple terms, we have two levels of branching. At the first level we branch on product type i.e. we set that a particular product type is always harvested in a given block (for the left branch) and the same product is never harvested (in the block under consideration) for the right branch. At the second level we assign supply of a product type to a sawmill, in one branch, and block the supply of the same product for the same sawmill in the other branch. Note that once we assigned a particular product to any block in a branch we always count this product in determining the harvesting cost, even if, in the branch-and-bound tree, we encounter the null component correspond to the same product type in a newly generated column. Following steps list the criteria we used to identify a product type and sawmill for branching:

1. Select a column (correspond to a basic variable) having maximum number of product types in a block such that at least one product is shared by at least one or more basic columns. Mathematically, let us define $\beta_{h,i,k,l}$ such that $\beta_{h,i,k,l} = 1$ if $z_i^h > 0$ and $x_{h,i,j}$ correspond to product type (k,l) is greater than zero, else $\beta_{h,i,k,l} = 0$, where $h \in H, \quad i \in G^h, \quad k \in K, \quad l \in L$, i.Now identify a pair i_1, h_1 such that

$$\sum_{k,l} \beta_{h_1,i_1,k,l} > \sum_{k,l} \beta_{h_2,i_2,k,l}$$

where $h_1 \neq h_2, h_1, h_2 \in H$, $i_1 \in G^{h_1}, \quad i_2 \in G^{h_2}, \quad z_{i_1}^{h_1} > 0, \quad z_{i_2}^{h_2} > 0$.
2. Introduce a constraint in the left branch to harvest the selected product in h_1 and a constraint to never harvest the product in h_1 for the right branch. Continue the branch-and-price procedure. If the constraint for the same product type and the same block already exits then go not next step.
3. Select a sawmill having largest number of columns (basic columns correspond to h_1) with positive component correspond to the selected product type. We refer this sawmill as preferred sawmill. Introduce a constraint to supply the selected product type from h_1 to the preferred sawmill in the left branch-and a constraint to bloc the supply of the same product type from h_1 to the sawmill in the right branch.

If we do not identify any product type or sawmill then the solution is integer. For a detailed overview of above approach, other branching strategies, and useful results reader can consult Savelsbergh (1997) and Barnhart et al. (1998).

The efficiency of column generation approach depends upon how effectively we solve the sub-problem. To solve $Pr(h)$ using integer programming is very costly. Since the sub-problem is dealing with only one block we can develop a fast algorithm to find a near optimal solution. Let $G(k,l)$ represent a set of indexes of all x which correspond to product type (k,l). Mathematically: $G(k,l) = \{f(k,l,s) \mid \quad s = 1,2,\ldots,S\}$. $C(G(k,l))$ gives the portion of cost associated with decision variables in $G(k,l)$ i.e. for a given h and χ

$$C(G(k,l)) = \sum_{j \in G(k,l)} (\beta j - b_{h,\chi}) \cdot x_j.$$

Now the algorithm is as follows:

Let $X^* = x_1^*, \quad x_2^*, \ldots, x_r^*$ is the set of decision variables to store the optimal solution. Define the sets $G(k,l)$ for all $k \in K$ and $l \in L$.

Algorithm 2

1. Set $\chi = 0$, and set BestCost = 0.0.
2. Set the $\chi = \chi + 1$ and solve the problem $P3$ (see Appendix). Let $X^1 = x_1^1, \ x_2^1, \ldots, x_r^1$ be the solution.
3. Define $G(k,l)$ for the solution and set counter=0.
4. For k=1 to K, and For $l = 1$ to L.
   (a) Counter = Counter + 1.
   (b) Compute cost[Counter] = $C(G(k,l))$.
5. Sort the array *cost*[] in decreasing order and set $sum = \sum_{i=1}^{\chi} cost[i]$. We keep in memory the indexes (k, l) corresponding to the first $1,\ldots,\chi$ *best cost after sorting. We represent it by* $(k^1,l^1),(k^2,l^2),\ldots,(k^\chi,l^\chi)$.
6. If sum < BestCost, stop, else
7. Set BestCost=sum.
8. For $i = 1,\ldots,\chi$
   (a) Set $x_j^* = x_j^1$ if $j \in G(k^i,l^i)$. Else
   (b) Set $x_j^* = 0$.
9. Go to 2.

6.1. Numerical example

n this section we present experimentation results for both the heuristic algorithm and the branch-and-price algorithm and compare with the optimal result obtained by Cplex. All the different problem instances are generated randomly using the following data. In all the examples, harvesting cost is randomly generated between 2 and 5 such that it should increase as production type increases. Similarly demand of each product type varies between 30 and 100, transportation cost varies between 1.0 and 5.0. In all cases we have fixed M equal to 250. The algorithms are coded in C++ and implemented on P4 700 MHz computer with 512 MB RAM.

Firstly, in Table 1, we present the computational performance of Algorithm 2. In all examples, we generated demand between 0 and 500 pieces, production cost for n product type is the sum of random number between 1 and 4 and the production cost of $n-1$ product types. Dual variables are generated between 0 and 70, transportation cost is generated according to the following formula: 0.1 * (unit product volume) + 0.01 * (index of sawmill). Ten instances of each problem size is solved. The mean percentage error and the percentage time gain *([Time taken by MIP − Time taken by heuristic]/[Time taken by MIP]).*

In Table 2, we compare B&P approach with heuristic (scenario improvement) and Optimal (MIP using Cplex). For small problem instances, less than 5 products and up to 4 stands, both MIP and heuristics performed very well but, Cplex takes less time compared to B&P. However, as the problem size grows, MIP consumes more time and memory resources. We can see that for bigger problems B&P approach outperformed the MIP using Cplex. In the Table 2, the first data corresponds to the solution obtained by the algorithm and the figure in bracket represents the time (in seconds) taken by the algorithm. For problem instances 15 and onwards (Table 2), we stop the MIP solver after 1000 seconds as beyond this time program was unable to continue because of memory limits. In brack-

Table 1
Computational performance (Algorithm 2)

Number of sawmills	Number of product types	Mean percentage error	Standard deviation of mean percentage error	Relative time saving in using algorithm 2 over MIP
5	15	2.15736	1.99564	0.68
5	20	2.09111	2.28346	0.84
5	25	3.01229	4.25334	0.85
5	30	1.98382	2.67125	0.98
10	15	3.0955	4.06499	0.86
10	20	0.513287	0.813698	0.93
10	25	2.34493	4.26834	0.95
10	30	0.547781	0.747188	0.99
15	15	2.32249	3.09511	0.92
15	20	0.871089	1.26744	0.95
15	25	2.19865	4.29367	0.97
15	30	0.607191	1.0062	0.99
20	15	1.51891	1.99983	0.95
20	20	1.59909	1.66634	0.97
20	25	1.46996	1.68545	0.98

Table 2
Computational performance

	Problem type (forest, sawmill, product)	MIP	Heuristic	Branch-and-price
1.	2, 2, 5	108.53 (0.16)	108.53 (0.078)	108.53 (2.54)
2.	2, 2, 5	98 (0.14)	98 (0.079)	98 (1.5)
3.	2, 2, 5	106 (0.06)	106 (0.015)	106 (1.42)
4.	2, 2, 5	105.3 (0.08)	105.3 (0.016)	105.3 (1.687)
5.	2, 2, 5	82.7 (0.01)	82.7 (0.031)	82.7 (0.92)
6.	4, 2, 5	87.502 (0.19)	87.5 (0.062)	87.5 (3.359)
7.	4, 2, 5	111.2 (0.14)	111.2 (0.265)	111.2(2.703)
8.	4, 2, 5	99.8 (0.2)	101.2 (0.062)	99.8 (2.96)
9.	4, 2, 5	92.38 (1.05)	94.98 (0.218)	92.38 (4.062)
10.	4, 2, 5	106.652 (0.19)	111.8 (0.093)	106.6 (3.4)
11.	4, 5, 5	259.41 (0.74)	259.41 (0.468)	259.41 (27.4)
12.	4, 5, 5	214.53 (3.77)	214.53 (0.5)	214.53 (22.92)
13.	4, 5, 5	290.79 (14.95)	294.06 (0.281)	290.79 (24.843)
14.	4, 5, 5	277.03 (10.67)	277.03 (0.203)	277.03 (20.156)
15.	4, 5, 5	287.94 (1.58)	293.386 (0.234)	287.94 (23.484)
16.	4, 5, 10	550.57 [10.05%]	562.51 (0.718)	552.6 (364.06)
17.	4, 5, 10	596.47 [8.2%]	634.97 (0.796)	596.43 (385.89)
18.	4, 5, 10	597.38 [7.57%]	644.27 (0.75)	595.7 (466.376)
19.	4, 5, 10	621.51 [6.89%]	631.13 (0.562)	624.508 (444.64)
20.	4, 5, 10	658.06[9.19%]	686.456 (0.515)	658.06 (412.127)

ets, we report the optimality gap given by Cplex at the end of 1000 seconds. We also terminate the branch-and-price approach when the optimality gap is less than or equal to 2.5%.

7. Conclusion

This work presents a problem of multi-commodity supply planning in the forest product supply chain. An integer programming formulation describes the complexity of the global problem. A branch-and-bound algorithm based on column generation approach is presented. A heuristic approach to find a good feasible solution in short time which also serves as an upper bound for the branch-and-bound algorithm is also developed. The approach is fast and provides a good solution quickly involving much less effort compare to the corresponding full scale problem formulation. The relative performance of the both algorithms are presented for the randomly generated practical size problems.

Acknowledgements

This research project has been carried out with the support of the Research Consortium FORAC of the Universit Laval, Qubec, Canada. The authors would like to thank Mr. Jean Favreau and Dr. Joseph Nader from the Forest Engineering Research Institute of Canada (FERIC) who have contributed to this research by their insightful comments and ideas.

Appendix

Here we present the formulation for calculating the cost c^h which is very close to the formulation P_1. For notation simplicity we drop the indexes h and t from $x_{h,t,f(l,k,s)}$ and introduce index n very similar to P_1 for identifying harvesting cost scenario

$$c^h = \sum_{n=1}^{N} \sum_{k\in K} \sum_{l\in L} \sum_{s\in S} (b_{h,n} + a^h_{l,s}) \cdot x_{f(l,k,s),n} \cdot d_{l,k,s}$$

under following constraints:

$$d_{l,k,s} \cdot x_{f(l,k,s),n} \leqslant Z \cdot J_{k,l} \quad \forall k \in K; \quad l \in L; \quad s \in S, \tag{21}$$

$$\sum_{l\in L} \sum_{k\in K} \sum_{s\in S} x_{f(l,k,s),n} \leqslant Z \cdot I_n \quad \forall n \in 1,2,\ldots,N, \tag{22}$$

$$\sum_{l\in L} \sum_{s\in S} d_{l,k,s} \cdot x_{f(l,k,s),n} \leqslant V_k, \quad \text{for} \quad k \in K, \tag{23}$$

$$\sum_{i=1}^{N} i \cdot I_i = \sum_{k\in K} \sum_{l\in L} J_{k,l}, \tag{24}$$

$$\sum_{i=1}^{N} I_i \leqslant 1, \tag{25}$$

$$I_n \in \{0,1\}, \quad n \in \{1,2,\ldots,N\}, \tag{26}$$

$$J_{l,k} \in \{0,1\} \quad \forall k \in K; \quad l \in L. \tag{27}$$

Combining the c^h function in problem $Pr(h)$ we find that the cost associated with variable $y_{h,t,h}$ is fixed (see the constraint which set the value 1) and the objective is to maximize the rest of the objective function. The objective function consist of decision variables for each scenario (scenario corresponding to every product type cost) and the dual variables. Combining the dual variables with the scenario cost (transportation cost and the production cost) we have $(u_{f(k,l,s)} - a^h_{l,s} - b_{h,n})$. We use the same expression in MIP formulation of pricing algorithm.

Let us define $\beta_{k,l,s} = u_{f(k,l,s)} - a^h_{l,s}$ now the problem $P3(\chi)$ is defined as follows:

$$P3(\chi)\ \text{Max} \sum_{k\in K} \sum_{l\in L} \sum_{s\in S} (\beta_{k,l,s} - b_{h,\chi}) \cdot x_{f(k,l,s)}$$

Subject to

$$d_{l,k,s} \cdot x_{f(k,l,s)} \leqslant V^h_k \quad \forall k \in K, \quad l \in L. \tag{28}$$

References

Aikens, C., 1985. Facility location models for distribution planning. European Journal of Operational Research 22, 263–279.
Antonio, J., Chauvet, F., Chu, C., Proth, J.M., 1999. The cutting stock problem with mixed objectives: Two heuristics based on dynamic programming. European Journal of Operational Research 114, 395–402.
Arce, E.J., Carnieri, C., Sanquetta, R.C., Filho, A.F., 2002. A forest-level bucking optimization system that considers customer's demand and transportation costs. Forest Science 48.
Barnhart, C., Johnson, E., Nemhauser, G., Savelsbergh, M., Vance, P., 1998. Branch-and-price: Column generation for solving huge integer programs. Operations Research 46.
Beaudoin, D., Lebel, L., Frayret, J.M., 2007. Tactical supply chain planning and robustness analysis in the forest products industry. Canadian Journal of Forest Research 37 (1), 128–140.
Boychuk, D., Martell, D., 1996. A multistage stochastic programming model for sustainable forest-level timber supply under risk of fire. Forest Science 42.
Bredström, D., Lundgren, J.T., Rönnqvist, M., Carlsson, D., Mason, A., 2004. Supply chain optimization in the pulp mill industry ip models, column generation and novel constraint branches". European Journal of Operational Research 156, 2–22.
Brunberg, T., Arlinger, J., 2001. Vad kostar det att sortera virket i skogen? (what does it cost to sort timber at the stump?). Resultat, 3. Available from: <http://www.skogforsk.se/upload/Dokument/Resultat/2001-03.pdf>.
Chauhan, S., Eremeev, A.V., Kolokolov, A.A., Servakh, V.V., 2002. On solving concave cost supply management problem with single manufacturing unit. In: Proceedings of SCM Conference, Poland.
Dantzig, G.B., Wolfe, P., 1960. Decomposition principles for linear programming. Operations Research 8, 101–111.
Epstein, R., Morales, R., Seron, J., Weintraub, A., 1999. Use of or systems in the chilean forest industries. Interfaces 29.
Hans, E., 2001. Resource loading by branch-and-price techniques. Ph.D. thesis. Beta Research School for Operations Management and Logistics.
Fisher, M., 1985. An application orientated guide to Langrangian relaxation. Interfaces 15 (2), 10–21.
Geoffrion, A., 1974. Langrangian Relaxation for Integer Programming. Mathematical Programming Study2: Approaches to Integer Programming Balinski, New York: North-Holland. pp. 82–114.
Gingras, J.-F., Favreau, J., 2002. Incidence du triage sur la productivitT des systFmes par bois trontonnTs. Avantage 3.
Hoffman, K.L., Padberg, M., 1985. LP-based combinatorial problem solving. Annals of Operations Research 4, 145–194.
Krarup, J., Pruzan, P., 1983. The simple plant location problem: Survey and synthesis. European Journal of Operational Research 12, 3681.
Laroze, A., 1999. A linear programming, tabu search method for solving forest-level bucking optimization problems. Forest Science 45.
Martell, D.L., Gunn, E.A., Weintraub, A., 1998. Forest management challenges for operational researchers. European Journal of Operational Research 104, 1–17.
Mitten, L., 1970. Branch-and-bound methods: General formulation and properties. Operations Research 5, 268–279.
Murty, K., 1988. Linear and Combinatorial Programming. John Wiley & Sons. ISBN: 0-471-57370-1.
Nemhauser, G., Wolsey, L., 1988. Integer and Combinatorial Optimization. Wiley-Interscience Publication.
Oliver, H., 2002. Decomposition approaches for solving the integer one-dimensional cutting stock problem with different type of standard lengths. European Journal of Operational Research 141, 295–312.
Rönnqvist, M., 2003. Optimization in forestry. Mathematical Programming, Series B 97.
Savelsbergh, M., 1997. A branch-and-price algorithm for the generalized assignment problem. Operations Research 45, 831–841.
Schneeweiss, C., 2003. Distributed Decision Making, second ed. Springer-Verlag, New York.
Umetani, S., Yagiura, M., Ibaraki, T., 2003. One-dimensional cutting stock problem to minimize the number of different patterns. European Journal of Operational Research 146, 388–402.
Vance, P.H., Barnhard, C., Johnson, E.L., Nemhauser, G.L., 1994. Solving binary cutting stock problems by column generation and branch-and-bound. Computational Optimization and Applications 3, 111–130.
Winston, W.L., 1993. Operations Research Application and Algorithms. Duxbury Press.