An Exact Best-First Search Procedure for the Constrained Rectangular Guillotine Knapsack Problem

K. V. Viswanathan and A. Bagchi

Indian Institute of Management Calcutta
P. O. Box 16757, Calcutta - 700 027, INDIA

Abstract : The Constrained Rectangular Guillotine Knapsack Problem (CRGKP) is a variant of the two-dimensional cutting stock problem. In the CRGKP, a stock rectangle of dimensions (L,W) is given. There are n different types of demanded rectangles, with the i^{th} type r_i having length l_i, width w_i, value v_i and demand constraint b_i. S must be cut using only orthogonal guillotine cuts to produce a_i copies of r_i, $1 \le i \le n$, so as to maximize $a_1v_1 + a_2v_2 + \ldots + a_nv_n$, subject to the constraints $a_i \le b_i$, $1 \le i \le n$. All parameters are integers. Here a new best-first search algorithm for the CRGKP is described. The heuristic estimate function is monotone, and optimal solutions are guaranteed. Computational results indicate that this method is superior in performance to the two existing algorithms for the problem.

I Introduction

Best-first search algorithms like A^* use heuristic estimates to direct search in large state spaces, as arise for example in solving puzzles such as the 15-puzzle. But not very many significant applications of best-first search to real-life problems are known. It is true that in many problems that are encountered in Operations Research, a search procedure must be employed to arrive at the best answer. The search typically involves visiting the nodes of an implicitly-specified tree, the order of traversal being determined by an evaluation function associated with the nodes. In a branch-and-bound formulation the search is often depth-first, the basic idea being to measure newly created nodes (potential solutions) against the best solution currently known, and to discard those found wanting. The method tends in general to be time consuming, since an essentially exhaustive search of the tree is undertaken to find the best solution. Best-first search can be viewed as a very special kind of branch-and-bound procedure where the search stops as soon as a complete solution (goal node) is found. Since heuristic estimates in practice are almost always admissible and generally monotone as well [Nilsson, 1980], an optimal solution is obtained. One expects a best-first search to run more quickly than a depth-first branch-and-bound procedure, but if both node expansion and heuristic computation are accomplished efficiently, a depth-first implementation can be very fast. This occurs with the well known method of [Little et al., 1963] for solving the travelling salesman problem. Even for the 15-puzzle, the modified depth-first algorithm called IDA^* [Korf, 1985] is quicker than A^*. Depth-first methods have the additional advantage that memory requirements are very low.

We describe here an application of best-first search to the Constrained Rectangular Guillotine Knapsack Problem (CRGKP), which is a variant of the two-dimensional cutting-stock (trim-loss) problem [Christofides and Whitlock, 1977], [Viswanathan, 1988], [Wang, 1983]. Christofides and Whitlock have described a depth-first branch-and-bound algorithm for the CRGKP which guarantees optimal solutions. Wang has studied a less general form of the CRGKP where values of the rectangles are proportional to their areas; his approach is heuristic, and solutions are not always optimal. No other algorithms for the CRGKP are known. Our method is superior to the abovementioned ones in that fewer nodes (rectangles) are generated, and the running time is smaller. The heuristic estimate function is monotone and optimal solutions are guaranteed. Cutting stock problems of one and two-dimensions arise in the glass, paper, steel, wood and other industries [Dyckhoff et al., 1985], and a close look at the CRGKP could help us to discover related problems which have efficient solutions using best-first search.

II Statement of the Problem

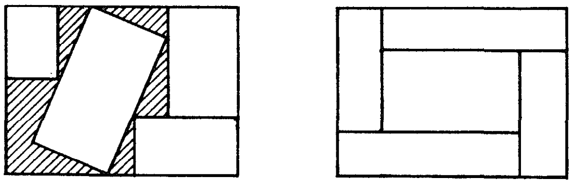
In the Constrained Rectangular Guillotine Knapsack Problem (CRGKP), we are given a single rectangular stock sheet S which must be cut in an optimal way into demanded rectangles of smaller size without violating specified constraints. All cuts must be orthogonal, i.e, parallel to one of the sides of S; moreover, any cut on S or on a rectangle obtained from S must be a guillotine cut, i.e, it must run from end to end on the rectangle. Fig 1(a) shows a non-orthogonal cutting pattern and Fig 1(b) a non-guillotine cutting pattern (shaded parts indicate waste). Formally, in the CRGKP, a stock sheet S of length L and width W is given. There are n types of demanded rectangles r_i, $1 \le i \le n$; the i^{th} type of demanded rectangle r_i has length l_i, width w_i, value v_i and demand constraint b_i. We are required to cut S using only guillotine cuts into a_i pieces of type i, $1 \le i \le n$, such that $Z = a_1v_1 + a_2v_2 + .. + a_nv_n$ is maximized, subject to the constraints $a_i \le b_i$,

$1 \leq i \leq n$. It is assumed that

i) L, W and l_i, w_i, v_i, b_i, $1 \leq i \leq n$, are all integers;

ii) the orientation of the rectangular pieces is fixed, i.e. a piece of length x and width y is not the same as a piece of length y and width x;

iii) all cuts on a rectangle are infinitesimally thin.



(a) (b)

Fig. 1

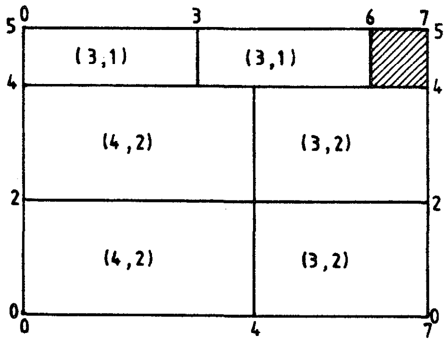


Fig. 2

Example 1 : Suppose L = 7, W = 5 and n = 3, and the other parameters are as given below :

i	l_i	w_i	v_i	b_i
1	3	1	10	3
2	3	2	15	2
3	4	2	25	3

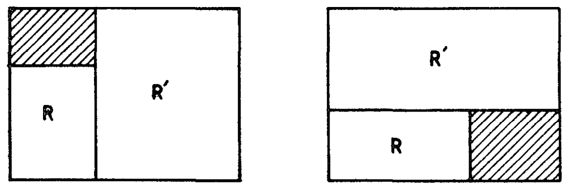
A solution to the problem, as shown in Fig 2, is

i	a_i	Z
1	2	
2	2	100
3	2	

In the figure, the shaded part indicates waste.

III Proposed Algorithm

The proposed algorithm BF_CRGKP can be viewed as resulting from a modification and extension of Wang's method. There are two lists, OPEN and CLIST. OPEN initially contains each of the n demanded rectangles r_i, $1 \leq i \leq n$. CLIST is initially empty. The evaluation function f assigns a total value f(R) to each rectangle R in OPEN; f(R) is the sum of the internal value g(R) and the heuristic estimate h(R). At each iteration, the rectangle R with the largest total value in OPEN is removed from OPEN and put in CLIST. Ties are resolved arbitrarily. New rectangles are then created from R by taking in turn each rectangle R' in CLIST (including R) and combining R and R' to form a horizontal build (see Fig 3(a)) and a vertical build (see Fig 3(b)). If the dimensions of R and R' do not match, a portion of a newly created rectangle will be waste. The new rectangles are put in OPEN and are thought of as the sons of R. Note however that a newly created rectangle Q is entered in OPEN only if Q has length $\leq$ L and width $\leq$ W (i.e only if Q fits into the stock rectangle), and Q satisfies the given demand constraints on the demanded rectangles; otherwise Q is just thrown away. The algorithm terminates when a rectangle R is selected from OPEN with heuristic h(R) = 0; then g(R), the internal value of R, is the optimal solution to the given instance of CRGKP.



(a) Horizontal build (b) Vertical build

Fig. 3

As mentioned above, f(R) = g(R) + h(R). The internal value g(R) of a rectangle R is simply the sum of the values v_i of each of the demanded rectangles that lie within R. To find h(R), we take the given stock rectangle S, and put R in the bottom left corner of S, as shown in Fig 4; we can then take h(R) to be some upper bound on the potential internal value of the portion P of S that lies outside R. A good upper bound can be found as follows : For the given demanded rectangles with their specified dimensions and values, let F(x,y) denote the optimal solution to the unconstrained rectangular guillotine knapsack problem for a stock rectangle of size (x,y). F(x,y) can be readily computed using the dynamic programming recursion of [Gilmore and Gomory, 1966]. Define the function $h_0(x,y)$ by the recursion

$$h_0(x,y) = \max \{ h_1(x,y), h_2(x,y) \},$$

$$h_1(x,y) = \max_{0 < u \leq L-x} \{ h_0(x+u,y) + F(u,y) \},$$

$$h_2(x,y) = \max_{0 < v \leq W-y} \{ h_0(x,y+v) + F(x,v) \},$$

$$h_0(L,W) = 0,$$

and let $h(R) = h_0(x_R, y_R)$

where (x_R, y_R) are the coordinates of the top right corner of R in Fig 4. However, it could happen that the additional demanded rectangles in P together with the demanded rectangles in R violate the given demand constraints on the demanded rectangles. This is why the evaluation function f gives only an upper bound on the optimal solution. To prevent the heuristic estimate from being misleading, we take the following additional precaution; if it is found that the introduction of even a single demanded rectangle whatsoever into P upsets the demand constraints, then we set $h(R) = 0$ even though $h_0(x_R, y_R) > 0$.

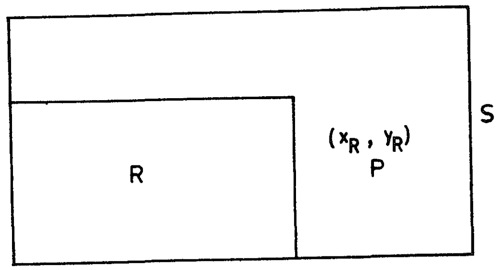


Fig. 4

The above computation for h_0 need only be done for those x which correspond to sums of multiples of lengths of the demanded rectangles, and for those y which correspond to sums of multiples of widths of the demanded rectangles. It is convenient and computationally feasible to tabulate the values of $h_0(x,y)$ in advance, so that when a new rectangle R is generated in OPEN during the execution of BF_CRGKP, only a table look-up is needed for assigning a value to h(R).

```
Algorithm BF CRGKP
begin
  OPEN := {r1,r2,....rn}; CLIST := empty set;
  finished := false;
  repeat
     choose a rectangle R from OPEN having
       highest total value;
     if h(R) = 0 then finished := true
     else begin
       transfer R from OPEN to CLIST;
       construct all guillotine rectangles Q
         such that
            i) Q is a horizontal or vertical
               build of R with some rectangle R' of
               CLIST,
           ii) dimensions of Q ≤ (L,W),
          iii) Q satisfies the demand constraints;
       put all newly constructed guillotine rec-
         tangles into OPEN with appropriate g, h,
         f values;
     end;
  until finished;
  output R;
end.
```

Example 2 : Consider the problem L = 5, W = 3, n = 2, and

i	l_i	w_i	v_i	b_i
1	2	2	25	1
2	3	1	10	5

The values of $h_0(x,y)$ are given in Table 1.

x	y	$h_0(x,y)$	x	y	$h_0(x,y)$
2	2	35	4	2	10
3	1	50	4	3	0
3	2	35	5	2	10
3	3	25	5	3	0

Table 1

Nodes (rectangles) are generated as shown in Fig 5. The root node corresponds to the null rectangle. Details about the generated rectangles are given in Table 2. Since ties can be resolved arbitrarily, we have assumed that nodes get selected from OPEN in the order

Rectangle R	1	2	3	6	7	8
f(R)	60	60	55	55	55	55

The solution obtained is shown in Fig 6(a). If we had chosen rectangle 9 instead of 8 at the end we would have obtained the same solution. The unconstrained optimum is shown in Fig 6(b), while the non-guillotine optimum is shown in Fig 6(c). Each row in Table 2 corresponds to a node (rectangle) in the tree of Fig 5. For each rectangle R, the table gives the number of occurrences of r_1 and r_2 in R, the length and width of R, whether R has been created by a horizontal (H) or vertical (V) build, and the values of g(R), h(R) and f(R). Note that rectangle 5 has a heuristic estimate of 0 because it is not possible to include a demanded rectangle in the remaining part of the stock rectangle S without violating the demand constraints on r_1.

Rect No.	r_1	r_2	length	width	V/H	g	h	f
1	1	0	2	2	–	25	35	60
2	0	1	3	1	–	10	50	60
3	0	2	3	2	V	20	35	55
4	1	1	5	2	H	35	10	45
5	1	1	3	3	V	35	0	35
6	1	2	5	2	H	45	10	55
7	0	3	3	3	V	30	25	55
8	1	3	5	3	V	55	0	55
9	1	3	5	3	H	55	0	55

Table 2

It can be shown formally that algorithm BF_CRGKP terminates and yields an optimal solution. We outline below the main steps in the proof.

A solution to the CRGKP specifies a guillotine cutting pattern, i.e a sequence of guillotine cuts on S and on rectangles obtained from S. Such cutting patterns have the following interesting property :

Theorem 1 : Any guillotine cutting pattern T_1 on S can be rearranged to get a new guillotine cutting pattern T_2 on S, such that any arbitrarily chosen rectangle in T_1, whether a demanded rectangle or a composite rectangle, is moved to the bottom left corner of T_2, and T_2 has the same composition of demanded rectangles as T_1.

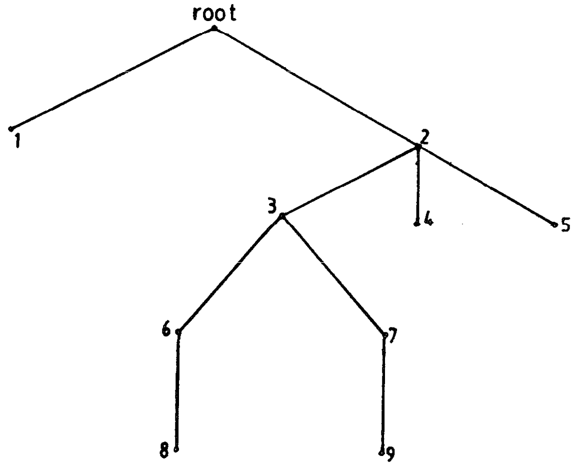
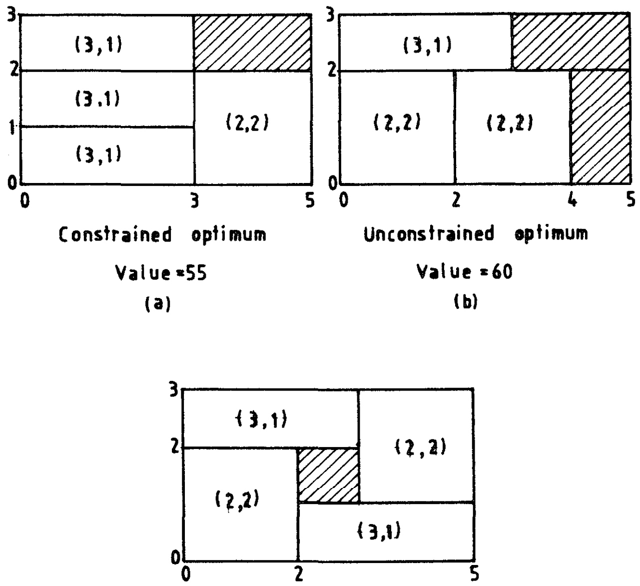


Fig. 5



Constrained optimum

Value = 55

(a)

Unconstrained optimum

Value = 60

(b)

Non guillotine optimum

Value = 70

(c)

Fig. 6

Theorem 1 motivates and clarifies our method of computing heuristic estimates. The next theorem formalizes the upper bounding property of the evaluation function f.

Theorem 2 : Let R be a rectangle generated in the course of an execution of Algorithm BF_CRGKP. Then $f(R) = g(R) + h(R)$ gives an upper bound on the maximum value obtainable from a guillotine cutting pattern that is constrained to include R.

Theorem 3 : The heuristic estimate function h is monotone.

Corollary : i) Let a rectangle R be a horizontal (or vertical) build of two rectangles R_1 and R_2. Then $f(R) \leq \min\{f(R_1), f(R_2)\}$.

ii) The time sequence of f-values of rectangles chosen from OPEN is non-increasing.

iii) At any time the f-value of a rectangle in CLIST is greater than or equal to the f-value of every rectangle of OPEN.

For a given instance of the CRGKP, let T be any guillotine cutting pattern that corresponds to an optimal solution. It should be observed that some component rectangle of the pattern T is in OPEN at each instance during the execution of BF_CRGKP. By our previous results we can then conclude that

Theorem 4 : Algorithm BF_CRGKP terminates and outputs an optimal solution.

Since Algorithm BF_CRGKP is a tree-search procedure, it is important to ensure that duplicate copies of rectangles are not generated. Duplication can cause an exponential explosion in the total number of nodes (rectangles) generated in the tree. Christofides and Whitlock have enumerated some sources of node duplication. Our implementation of BF_CRGKP incorporates checks to ensure that node duplication is cut down to a minimum. Details can be found in [Viswanathan, 1988].

IV Computational Results

Christofides and Whitlock give details on three test problems. Algorithm BF_CRGKP was run on these problems for purposes of comparison. The results are shown in Table 3. Running times are not given for the following reason. Christofides and Whitlock had implemented their algorithm in FORTRAN IV on the CDC 7600. BF_CRGKP was programmed in Pascal and run on the VAX-11/750. We also ran Christofides and Whitlock's algorithm in Pascal on the VAX-11/750, but although correct answers were obtained on the test problems, the number of nodes generated did not tally with those reported by Christofides and Whitlock and running times were orders of magnitude greater than for BF_CRGKP.

Wang's method was also programmed in Pascal on the VAX-11/750. For different stock sizes, a number of test problems were randomly generated using a scheme described by Christofides and Whitlock. Unfortunately, Wang's method being heuristic in nature does not yield optimal solutions in a single invocation. Using one of his suggestions it is possible, as a general rule, to get optimal solutions in two invocations. Table 4 gives the comparative running times for obtaining optimal output.

The heuristic estimate function h described here is not the only possible heuristic that can be used in BF_CRGKP. For details on other heuristic estimate functions see [Viswanathan, 1988].

No	Size of stock rectangle (L,W)	Number of demanded rects	Christofides and Whitlock's method : number of nodes in tree As reported	As obtained by us	BF_CRGKP Number of nodes in tree
1	(15,10)	7	3,794	49,638	498
2	(40,70)	10	18,602	39,308	4,110
3	(40,70)	20	22,184	116,550	14,936

Table 3

No	Stock Size (L,W)	Number of demanded rects	Number of problems solved	Wang's Method Avg no of rectangles	Wang's Method Avg CPU Time	BF_CRGKP Avg no of rectangles	BF_CRGKP Avg CPU Time
1	(40,70)	5	4	678	32.23	669	16.21
2	(53,65)	5	4	920	106.91	178	9.88
3	(50,100)	5	4	549	23.80	114	20.53
4	(15,10)	6	4	515	26.65	395	1.54
5	(40,70)	7	4	599	26.35	376	10.91
6	(40,70)	10	4	1221	135.11	1251	16.30

Table 4

V Conclusion

Cutting stock problems arise often in industry, and various interesting techniques have been devised for solving them (see for example [Gilmore and Gomory, 1961]). Many variants of one and two-dimensional cutting stock problems have been studied. This paper has been concerned with the Constrained Rectangular Guillotine Knapsack Problem (CRGKP). A convenient dynamic programming formulation for the unconstrained version of the problem is known, but the CRGKP calls for a more elaborate procedure. We have described a best-first algorithm for the CRGKP which appears to be superior to earlier methods. The significance of the algorithm lies in the fact that not too many successful applications of best-first search to real-life problems are known. For many tree-search problems, depth-first methods have been devised which run faster than best-first methods. Under what conditions is a best-first approach likely to prevail over a depth-first one ? It would seem that the problem must be such that the total time taken to expand a node, i.e the time taken to

i) generate the sons of the node, and

ii) to compute heuristic estimates for the sons

cannot be made too small. In the 15-puzzle, or in the method proposed by Little et al. for the travelling salesman problem, it is possible to reduce this time to such a small value that repeated node expansion becomes a feasible alternative. But not so for our problem. In BF_CRGKP, the heuristic estimate computation is essentially a table look-up, but the generation of sons of a node takes significant time; moreover, the algorithm does not have a convenient depth-first formulation. Is it possible to categorize the class of tree-search problems for which best-first implementations are preferable to depth-first ones ?

VI References

[Christofides and Whitlock, 1977] N. Christofides and C. Whitlock. An algorithm for two dimensional cutting problems, Operations Research, Vol 25, No 1, 1977, pp 30-44.

[Dyckhoff et al., 1985] H. Dyckhoff, H. J. Kruse, D. Abel and T.Gal. Trim loss and related problems, OMEGA, Vol 13, No 1, 1985, pp 59-72.

[Gilmore and Gomory, 1961] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem, Operations Research, Vol 9, 1961, pp 849-859.

[Gilmore and Gomory, 1966] P. C. Gilmore and R. E Gomory. The theory and computation of knapsack functions, Operations Research, Vol 14, 1966, pp 1045-1074.

[Korf, 1985] R. E. Korf. Depth-first iterative deepening : an optimal admissible tree search, Artificial Intelligence, Vol 27, 1985, pp 97-109.

[Little et al., 1963] J. D. C. Little, K. G. Murty, D. W. Sweeney and C. Karel, An algorithm for the travelling salesman problem, Operations Research, Vol 11, 1963, pp 972-989.

[Nilsson, 1980] N. J. Nilsson. Principles of Artificial Intelligence, Tioga-Springer-Verlag, Palo Alto, Calif., 1980.

[Viswanathan, 1988] K. V. Viswanathan. New Algorithms for Constrained Rectangular Guillotine Knapsack Problems, Fellow Programme Thesis, I. I. M Calcutta, January, 1988.

[Wang, 1983] P. Y. Wang. Two algorithms for constrained two-dimensional cutting-stock problems, Operations Research, Vol 31, No 3, 1983, pp 573-586.