A dynamic list heuristic for 2D-cutting

Luiz Antonio N. Lorena, Fábio Belo Lopes

INPE - Instituto de Pesquisas Espaciais

*LAC - Laboratório Associado de Computação e Matemática
Aplicada, Caixa Postal 515*

12201 - São José dos Campos - SP - Brazil

Abstract

We present a new heuristic for the 2D-cutting problem, called Dynamic List Heuristic (DLH). The objective is to eliminate the combinatorial explosion of the all rectangle combination's type heuristics, maintaining their good results of reduced waste. Algorithm DLH uses a dynamic list of constructed patterns, included and deleted at convenient times. DLH is based on the A* approach, a best-first tree of candidate solutions, and the use of an auxiliary tabu list for alternate best patterns to avoid excessive repetition. The computational tests for some problems of the literature and many random generated problems, confirm good results using microcomputers.

Keywords

Cutting Stock, Heuristics, Best-first search, Tabu search.

1 INTRODUCTION

The 2D-cutting problem received great attention in the cutting stock problem context. It is a problem of combining a limited demand of given rectangles to fit in a plate with smallest total waste. Dowsland and Dowsland (1992) and Israni and Sanders (1985) give surveys including 2D-cutting problems. For an annotated bibliography see Sweeney and Paternoster (1992).

Wang (1983) proposed two heuristics for the problem. The good results obtained are consequence of a combinatorial explosive sequence of horizontal or vertical combinations of rectangles. Its indication is only for small problems with medium rectangles. Trying to improve Wang algorithms, Oliveira and Ferreira (1990), used bounds of the unlimited demand heuristic

to improve the Wang's pruning criteria (algorithm one). The paper accelerates Wang algorithm one, maintaining all their original characteristics.

We present in this paper a new heuristic. We not guarantee optimal solutions, but a fast approximate algorithm for large scale problems, using microcomputer. It appears definitively indicated to use in microcomputers. In section two we present the motivation and describe the algorithm DLH. Section three presents computational tests for problems of the literature and randomly generated test problems. Finally we conclude with suggestions in section four.

2 HEURISTIC DLH

We begin defining rectangle constructions and the type of wastes resulted. Given a plate with dimensions W (width) and H (height), and primary rectangle's $r_1, r_2, \dots, r_m$, with bound limits $b_1, b_2, \dots, b_m$. L is a list of rectangles, initially composed by primary rectangles. A rectangle $R_i \in L$, fitting in the plate, is constructed by horizontal or vertical build of two other rectangles, primary and/or another rectangle $R_j \in L$ already constructed. Two constructed rectangles are equivalent when formed by the same primary rectangles and have the same waste.

Denote the internal waste in R_i by $I_w(R_i)$. The difference $(WH - R_i)$ is the external waste of R_i . The final pattern will be of guillotined type, and the objective is construct rectangles R_i minimizing total waste (internal + external).

Wang (1983) proposes two algorithms based in the following increasing power of pruning criteria for R_i :

$$I_w(R_i) > \beta_1 WH, \text{ for a given } \beta_1, 0 \leq \beta_1 \leq 1, \text{ and} \quad (1)$$

$$I_w(R_i) > \beta_2 R_i, \text{ for a given } \beta_2, 0 \leq \beta_2 \leq 1. \quad (2)$$

The list L of R_i 's increases combining recursively each two existing rectangle, that not satisfies pruning criteria (1) (algorithm one) or pruning criteria (2) (algorithm two), satisfies the bound limits for the primary rectangles, and avoids equivalent rectangles. For small primary rectangles, the number of combinations (patterns) fitting in the plate can be combinatorially explosive. The indications of Wang algorithms are only for small problems with medium/large primary rectangles.

Heuristic DLH is an A* like algorithm (1984). For a given R_i , defines a function $f(\alpha, R_i) = I_w(R_i) + \alpha(WH - R_i)$, $\alpha \geq 0$. The parameter α is a percentage of external waste that, along which the internal waste $I_w(R_i)$, constitute the expected total waste. That is, for a given β , $0 \leq \beta \leq 1$, the expected total waste is βWH and it is desirable that $f(\alpha, R_i) \leq \beta WH$. Interpreting in the A* context, the function f is a node-selection function, consisting of two parts, g and h , where $g(R_i) = I_w(R_i)$, is the current waste, and $h(\alpha, R_i) = \alpha(WH - R_i)$ is an estimate of the total waste.

The new pruning criteria for R_i is

$$I_w(R_i) + \alpha(WH - R_i) > \beta WH. \quad (3)$$

For $\alpha = 0$, $I_w(R_i) > \beta WH$, reproduces pruning criteria (1) for $\beta = \beta_1$, and for $\alpha = \beta$, $I_w(R_i) > \beta R_i$, reproduces pruning criteria (2) for $\beta = \beta_2$.

Heuristic DLH makes a tree with a new R_i for each node. Build the nodes by the rules governing the construction of rectangles, that is, avoiding the pruning criteria (3) and equivalent rectangles, and satisfying the bound limits for primary rectangles. For the new list L , instead of combines each two existing rectangles to construct new rectangles, the rectangle corresponding to the minimum $f(\alpha, R_i)$ can be combined with the other existing rectangles, characterizing a best-first search. The parameter α must be (smoothly) updated at each iteration, beginning small, $\alpha < \beta$, and increasing as the external waste decreases. When α increases, the pruning criteria (3) increase in power.

These are basic lines for the algorithm design, but some points must be carefully explained, such that:

- When α increases some candidate patterns of L are deleted characterizing a dynamic list of patterns;
- The "good" candidate patterns must remains in L to continue the combinations, at least for a "convenient" time;
- The stopping criteria can be when the list L is empty, where the best pattern (minimum total waste) was retained;
- The primary rectangles are also pruned for some values of α , so these values must only be reached after some time to improve the list L ;
- For all $R_i \in L$ the function $f(\alpha, R_i)$ changes with α , and must be recalculated, what seems to be inefficient for implementation purposes.

For given β , $0 \leq \beta \leq 1$ and $R_i \in L$, the auxiliary parameters γ_i , δ_i , and ε_i are defined as

$$\gamma_i = I_w(R_i)/\beta WH,$$

$$\delta_i = R_i / WH, \text{ and}$$

$$\varepsilon_i = (1 - \gamma_i)/(1 - \delta_i).$$

Parameter γ_i , $0 \leq \gamma_i \leq 1$, is the proportion of total waste reached by R_i , and δ_i , $0 \leq \delta_i < 1$, the proportion of total area occupied by R_i . The pruning criteria (3) is then equivalent to

$$\alpha/\beta > \varepsilon_i. \quad (4)$$

If $\delta_i = 1$ or $R_i = WH$, $\varepsilon_i \rightarrow \infty$, and we arrive in a possible optimal solution for that β . To continue the algorithm and search better solutions for updated β it is necessary set $\delta_i = 0.99$ in this case.

Let $\varepsilon_{sp} = \min \{\varepsilon_i\}$, for ε_i corresponding to primary rectangles, that is $\varepsilon_i = 1/(1-\delta_i)$. It is immediate that $\varepsilon_{sp} > 1$. For all $\varepsilon_i \in [0, 1]$, $\gamma_i \geq 1 - \varepsilon_i$ and for all $\varepsilon_i \in (1, 2]$, $\delta_i \geq 1/\varepsilon_i$.

Given α and β , DLH maintain two kind of lists. An ordered list L of possible candidate solutions R_i , and a list F for temporary constructed solutions, subsequently incorporated in L . The list L is ordered considering a non-decreasing order of $|\varepsilon_{sp} - \varepsilon_i|$. At each step, criteria (4) possibly delete some candidate solutions of L , the rectangle R_s with minimum total waste $[I_w(R_s) + (WH - R_s)]$ is retained, and if $\beta WH > [I_w(R_s) + (WH - R_s)]$, then β can be updated as $\beta = [I_w(R_s) + (WH - R_s)]/WH$, and R_s is a possible optimal solution.

The best-first part of DLH is the selection of the index p corresponding to the minimum $\{|\varepsilon_{sp} - \varepsilon_i|\}$, and combining R_p with the existing rectangles in L for construction the new rectangles of F . The original proposition was to use the index of minimum $f(\alpha, R_i)$, but as ε_i is independent of α , its use is more efficient for implementation, because α changes at each iteration, and the ε_i 's change only with β changes. The first F is formed combining the smallest primary rectangle with the other ones. Note that $\varepsilon_{sp} > 1$, and primary rectangles can be pruned only for $\alpha \geq \beta$. Using $|\varepsilon_{sp} - \varepsilon_i|$ primary rectangles are R_p even for $\alpha < \beta$, a characteristic that accelerates the method. When $\alpha = 2\beta$, all rectangles with $\varepsilon_i < 2$ are pruned or equivalently all rectangles with area $< 50\%$ of the total area are pruned, and the rectangles remaining can not be candidate for new combinations. Heuristic DLH stops for $\alpha \geq 2\beta$ or for an empty list L , with $\{R_s\}$ and a possibly updated β . To permit a more generalized selection of index p and avoid excessive repetition, a tabu list of the last five candidates is maintained. If an index is choosed to be the index p at current iteration, only after five iterations it can be reconsidered to be the index p again.

Parameter α is in the kernel of DLH. When it increases, some patterns are deleted of L , and good solutions can be lost in the process. Then, parameter α must be carefully updated. Parameter ε_i also gives some insight in controlling α . For primary rectangles, $I_w(r_i) = 0$, $i = 1, \dots, m$, and consequently $\varepsilon_i = 1/(1 - \delta_i)$. Using criteria (4), primary rectangles up to 50% of the total area, are pruned for $\beta < \alpha < 2\beta$. Then it is interesting a smoothly increasing of α between β and 2β , providing some more time for combining primary rectangles.

For primary rectangles we have used the smallest ε_i (ε_{sp}) for selection of the candidate rectangle for combinations. Parameter ε_{sp} and the medium ε_i ($\varepsilon_{mp} = \sum \varepsilon_i / m$) and the greater ε_i ($\varepsilon_{gp} = \max \{\varepsilon_i\}$) can be of value to determine the step size for α . As a suggestion, for a given β , α can be a function of β , setting

$$\begin{aligned}\alpha &:= \alpha + (0.01)\beta, \text{ for } \alpha < [\varepsilon_{sp} - 0.35(\varepsilon_{mp} - \varepsilon_{sp})]\beta, \\ \alpha &:= \alpha + [(\varepsilon_{mp} - \varepsilon_{sp})/300]\beta, \text{ for } [\varepsilon_{sp} - 0.35(\varepsilon_{mp} - \varepsilon_{sp})]\beta < \alpha \leq \varepsilon_{mp}\beta, \\ \alpha &:= \alpha + [(\varepsilon_{gp} - \varepsilon_{mp})/300]\beta, \text{ for } \varepsilon_{mp}\beta < \alpha \leq [\varepsilon_{gp} + 0.35(\varepsilon_{gp} - \varepsilon_{mp})]\beta, \\ \alpha &:= \alpha + (0.01)\beta, \text{ for } \alpha > [\varepsilon_{gp} + 0.35(\varepsilon_{gp} - \varepsilon_{mp})]\beta.\end{aligned}$$

For an initial $\alpha = 0$, DLH perform $[\varepsilon_{sp} - 0.35(\varepsilon_{mp} - \varepsilon_{sp})]/(0.01)$ iterations with $\alpha < [\varepsilon_{sp} - 0.35(\varepsilon_{mp} - \varepsilon_{sp})]\beta$, and no primary rectangles are pruned. For the next iterations, until $\varepsilon_{mp}\beta$, the tolerance factor $0.35(\varepsilon_{mp} - \varepsilon_{sp})$ avoid immediate pruning of the smallest primary rectangle, and the step size $(\varepsilon_{mp} - \varepsilon_{sp})/300$ is considerably decreased, because $(\varepsilon_{mp} - \varepsilon_{sp}) < 1$ for a "normal" ($\varepsilon_{mp} < 2$) set of problems. As the zone $\varepsilon_{sp} \pm \Delta$ is of desirable rectangles, the smoothly variation is beginning. After $\varepsilon_{mp}\beta$, the new step size $(\varepsilon_{gp} - \varepsilon_{mp})/300$ determine the variation. In general, for "normal" set of primary rectangles we also have $(\varepsilon_{gp} - \varepsilon_{mp}) < 1$ and the smoothly variation continues. The final iterations are conducted with the first step size (0.01).

We examine in the following a formal description of DLH:

Heuristic DLH**Begin**

Choose a value for β , $0 \leq \beta \leq 1$ and for α , $0 \leq \alpha \leq \beta$;

Given $W, H; r_1, r_2, \dots, r_m;$
 $b_1, b_2, \dots, b_m;$
 $\text{total_waste} := WH;$
 $\text{mult1} := 0.01;$

For $i = 1$ to m **do**

$R_i := r_i;$
 $\delta_i := R_i / WH;$
 $\epsilon_i := 1/(1-\delta_i);$
 $\text{waste} := WR - R_i;$
if $\text{waste} < \text{total_waste}$ **then**
 $\text{total_waste} := \text{waste};$
 $s := i;$

end_if

end_for

Compute $\epsilon_{sp} := \min \{\epsilon_i\};$
 $\epsilon_{mp} := \sum \epsilon_i / m;$
 $\epsilon_{gp} := \max \{\epsilon_i\};$
 $\text{mult2} := (\epsilon_{mp} - \epsilon_{sp})/300;$
 $\text{mult3} := (\epsilon_{gp} - \epsilon_{mp})/300;$
 $\text{factor1} := \epsilon_{sp} - 0.35(\epsilon_{mp} - \epsilon_{sp});$
 $\text{factor2} := \epsilon_{gp} + 0.35(\epsilon_{gp} - \epsilon_{mp});$

end_compute

Define initial list $L(0) = \{R_1, R_2, \dots, R_m\};$

$k := 1; l := 0; n := m;$

While $L(k-1)$ is non empty and $\alpha < 2\beta$ **do**

$t := 0; n := n + l; l := 0;$

for $i = 1$ to n **do**

if $\alpha/\beta > \epsilon_i$ **then**
 $L(k-1) := L(k-1) - \{R_i\};$
 $t := t + 1;$

end_if

end_for

$n := n - t;$

Compute

$p := \text{index of tabu list for}$
 $\min \{|\epsilon_{sp} - \epsilon_i|\};$

$F(k)$ which is the set of l rectangles

$R_j, j = 1, \dots, l$, satisfying

(i) each R_j is formed by an horizontal or vertical build of rectangle R_p and a rectangle from $L(k-1)$;

(ii) $\alpha/\beta \leq \epsilon_j = (1 - \gamma_j)/(1 - \delta_j)$, where
 $\gamma_j := I_w(R_j)/\beta WH$ and $\delta_j := R_j/WH$, for
 $R_j \neq WH$ and $\delta_j = 0.99$ for $R_j = WH$,
 $j = 1, \dots, l$;

(iii) those rectangles $r_i, i = 1, \dots, m$, appearing in R_j do not violate the bound constraints $b_1, b_2, \dots, b_m$;

end_compute

for $i = 1$ to l **do**

$\text{waste} := (WH - R_i) + I_w(R_i);$

if $\text{waste} < \text{total_waste}$ **then**

$\text{total_waste} := \text{waste};$

$s := i;$

end_if

end_for

If $\beta\epsilon_s > 1$ **then**

$\beta := [I_w(R_s) + (WH - R_s)]/WH;$

for $i = 1$ to n **do**

$\gamma_i := I_w(R_i)/[I_w(R_s) + (WH - R_s)];$

$\epsilon_i := (1 - \gamma_i)/(1 - \delta_i);$

end_for

end_if

$L(k) = L(k-1) \cup F(k)$ and **remove** any equivalent rectangle from $L(k)$;

$k := k + 1;$

If $\alpha < (\text{factor1})\beta$ **then** $\alpha := \alpha + (\text{mult1})\beta$

else if $\alpha < \epsilon_{mp}\beta$ **then** $\alpha := \alpha + (\text{mult2})\beta$

else if $\alpha < (\text{factor2})\beta$

then $\alpha := \alpha + (\text{mult3})\beta$

else $\alpha := \alpha + (\text{mult1})\beta$

end_if

end_while

Write $\{R_s\}$, total_waste and β ;

end

3 COMPUTATIONAL TESTS

The Algorithm DHL was tested on some problems of the literature and a number of randomly generated test problems. The problems of the literature are from Wang (1983), three from Christofides and Whitlock (1977) (with no values assigned to rectangles), and two from Oliveira and Ferreira (1990). The generated problems can be seen from Table 1.

Table 1: Random generated problems

Problem Set	Area		Aspect Ratio		Bound Limits		m	No. of Problems
	mean	σ	mean	σ	mean	σ		
P10	9%	20%	1	80%	2	60%	10	3
P20	9%	20%	1	80%	2	60%	20	3
P50	9%	20%	1	80%	2	60%	50	3
P100	9%	20%	1	80%	2	60%	100	3
P200	9%	20%	1	80%	2	60%	200	3
P200.2	2%	20%	1	80%	2	60%	200	1

The method used to generate the random problem is given by the following expressions:

$$\text{Area}(r_i) = N(A, \text{std}A)WH$$

$$\text{Width}(r_i) = \sqrt{N(\text{Asp}, \text{stdAsp}) \cdot \text{Area}(r_i)}$$

$$\text{Height}(r_i) = \text{Area}(r_i) / \text{Width}(r_i)$$

$$b_i = N(\text{BL}, \text{stdBL})$$

where

$\text{Area}(r_i)$, $\text{Width}(r_i)$ and $\text{Height}(r_i)$: area width and height of primary rectangle i , respectively.

$N(m, \sigma)$: normal distribution with mean m and standard deviation σ .

A , $\text{std}A$: average area and the standard deviation of the rectangles, respectively.

Asp , stdAsp : average aspect ratio (width / height) of the rectangles and its standard deviation, respectively.

BL , stdBL : average bound limits and its standard deviation, respectively.

Some cares were taken to assure that the rectangles do not exceed the plate width or height. Finally, the plate size was 70×40 for all problems; the number of different rectangles ranged from 10 to 200, and we assume that the rectangles were of fixed orientation, although it is not a restriction to the algorithm.

The results were obtained by coding the algorithm DLH in C (Turbo C 2.0), and executing on an IBM-PC 486 computer working at 33 MHz, under MS-DOS 6.0.

Since DLH forms a new rectangle by a horizontal *or* vertical build, it is somewhat random. Therefore, the program was run five times for each test problem. The results are shown in Table 2. For all test problems, $\alpha = 0$ and $\beta = 0.08$ at the start of DLH.

Some comments about Table 2:

- All wastes are smaller than 10% of plate size, except in one case, the problem P2200. For this problem we reached good results halving the plate. The cut can be horizontal (P2200⁽¹⁾) or vertical (P2200⁽²⁾), and the best wastes are reduced to 7% and 10%, respectively.
- The number of partial solutions generated by DLH is smaller than one generated by the algorithms of Wang (1983) and Oliveira and Ferreira (1990), for problems of similar size.

Table 2: Computational results.

Problem	Waste		Average Time (sec)	Maximum Size of L	No. of partial solutions
	Min.	Max.			
Wang	100	100	16.21	228	911.8
Chris1	9	10	12.50	147	372.8
Chris2	163	278	47.57	574	1469.8
Chris3	100	127	15.49	228	723.8
Olive1	87	140	17.14	218	808.2
Olive2	110	278	12.75	165	540.6
P10-1	80	106	21.15	409	1255.4
P10-2	60	144	21.34	405	812.8
P10-3	124	136	11.62	265	444.8
P20-1	112	149	63.01	1571	2066.8
P20-2	163	184	138.44	1723	4837.0
P20-3	154	169	123.56	1629	4712.4
P50-1	126	172	201.30	1611	6950.2
P50-2	138	180	201.81	1640	7213.0
P50-3	166	198	219.53	1651	8369.4
P100-1	159	200	305.11	1654	11197.6
P100-2	126	184	349.33	1888	13209.8
P100-3	133	160	337.47	1783	12398.8
P200-1	117	165	375.15	2003	12327.0
P200-2	117	171	334.07	1669	10570.0
P200-3	141	189	393.83	1882	13574.4
P2200	1063	1349	629.08	2479	21216.4
P2200 ⁽¹⁾	99	122	575.13	2075	22625.2
P2200 ⁽²⁾	136	207	482.26	2014	22851.6

(1) Plate of 70x20

(2) Plate of 35x40

The tests in table 2 was obtained with a fixed value of β . Varying this value better results can be obtained. For example, the best results for the problems of the literature are presented in table 3, with the referred β values. All but one optimal solutions are obtained.

Table 3: Computational results varying β .

Problems	Optimal solution	β	Waste	Times (sec.)	Max. size of L	Number of partial solutions
Wang	79	0.25	79	128	1314	3400
Chris1	6	0.07	6	65	129	480
Chris2	22	0.05	60	76	346	630
Chris3	79	0.25	79	128	1314	3400
Olive1	63	0.06	63	60	83	235
Olive2	110	0.06	110	64	349	121

4 CONCLUSION

We proposed in this paper a new heuristic for the 2D-cutting problem. Heuristic DLH performs a best-first search in a tree of candidate solutions, builded by horizontal or vertical combinations of rectangles. The main characteristic is the use of a dynamic list of candidate patterns, pruned by dynamic pruning criteria, getting the essence of other Wang's based heuristics, while eliminate combinatorial explosion.

The good results obtained in section three can be improved if we admit rotations for rectangles. For the case of small rectangles (2% of plate), we can improve the criteria for the choice of the first cut. The case of selection of plates can be directly extended solving a knapsack area based, considering a tolerance for waste, at least for the case of greater demand.

REFERENCES

- Christofides, N., and Whitlock, C. (1977), An algorithm for two-dimensional cutting problems, *Operations Research* **25**, 31-44.
- Dowsland, K.A., and Dowsland, W.B. (1992), Packing problems, *European Journal of Operational Research* **56/2**, 2-14.
- Israni, S.S., and Sanders, J.L. (1985), Two-dimensional cutting stock problem research: a review and a new rectangular layout algorithm, *J. Manufacturing Sys.* **1**, 169-182.
- Oliveira, J.F., and Ferreira, J.S. (1990), An improved version of Wang's algorithm for two-dimensional cutting problems, *European Journal of Operational Research* **44**, 256-266.
- Pearl, J. (1984), *Heuristics: intelligent search strategies for computer problem solving*, Addison-Wesley Publishing Co., London.
- Sweeney, P.E., and Paternoster, E.R. (1992), Cutting and packing problems: a categorized, application-orientated research bibliography, *Journal of the Operational Research Society* **43/7**, 691-706.
- Wang, P.Y. (1983), Two algorithms for constrained two-dimensional cutting stock problems, *Operations Research* **31**, 573-586.