ORiON, Vol. 3, No. 1, pp. 42-55 *ISSN 0259-191X*

CUTTING PAPER IN RICHARDS BAY:

DYNAMIC LOCAL OR GLOBAL OPTIMIZATION IN THE TRIM PROBLEM?

Adolf Diegel, Professor of Decision Science
Faculty of Commerce, University of Natal, Durban 4001

Abstract

The Cutting Stock or Trim Problem arises when jumbo rolls of paper are slit into reels of various width ("deckled"). The problem was examined as early as 1954 [1]; it is now seen as a classical Linear Programming formulation [2] [4] [6] [7].

The problem may be easy to formulate, but it is difficult to solve for genuine data. This is so not only because of the computations required to find good cutting plans. The major problem is this: such plans are based on certain data, but the data change as the plan is being executed and the plan no longer applies. So here it may not be optimal to optimize globally, but to work on a local heuristic basis [3].

We can propose a heuristic procedure that compares well with an absolute solution and that can be used on a local basis. This heuristic is in use at Mondi Paper in Richards Bay [5].

Introduction

Mondi Paper in Richards Bay can cut about 40 jumbos per day. A set of orders for a given type of paper may have hundreds of reels in many different sizes. For example, here are the data that were first given to the author as a typical set:

Table 1. Current orders for a given type of paper

Size	No.	Size	No.	Size	No.	Size	No.	Size	No.
2465	37	2440	6	2385	3	2330	8	2305	4
2235	3	2185	7	2175	10	2160	10	2135	22
2070	31	2035	14	2030	13	2025	5	2020	21
1990	15	1980	16	1955	7	1950	13	1930	10
1920	15	1900	8	1860	23	1850	7	1830	6
1800	9	1780	3	1745	6	1720	10	1675	7
1665	4	1640	8	1600	3	1560	3	1550	20
1525	3	1475	4	1430	2	1380	48	1360	10
1350	7	1235	9	1225	12	1140	9	1130	7
1090	6	1055	2	1015	2	990	5	975	9
965	2	935	2	920	38	845	13		

The size in question is the width of a reel of paper wanted by a client. The number means the total number of reels of that size which are currently on order, possibly for several clients. All in all we have 54 different sizes (ranging from 845 mm to 2465 mm). The total number of reels comes to 577.

Some sizes require as few as 2 reels, but for others we want as many as 37 or 38, even 48. So how are we going to match such different sizes and numbers to fill a jumbo? And how many jumbos are needed to fill this set of orders?

Lowest possible number of jumbos

How many jumbos are needed? The answer depends on whether we take the question theoretically or practically.

The theoretical answer assumes that all jumbos are deckled perfectly, without any waste. So find the length covered by all reels, and divide the total by the length of a jumbo.

Here we want 37 reels of 2465, 6 of 2440, ..., plus finally 13 of 845: the total length of all reels is 997305 mm. You cover that distance by placing all the 577 reels end to end.

A standard jumbo is 4800 mm, and 997305/4800 makes 208: that many units cover the distance if again the jumbos are placed end to end (disregarding the fact that the jumbo ends do not match the reel ends). So 208 is the absolutely lowest number of jumbos needed for our data, without any waste.

Absolute (global) optimization

How many jumbos are used in fact? That depends on how well we match the reel sizes that go on a given jumbo.

Ideally we do so through the classical LP formulation of our problem. It enumerates all possible combinations of reels that fit on a jumbo, then selects the patterns filling the demand at hand with the fewest jumbos.

This solution is well illustrated in any of our references, so it may seem superfluous to present a full example. Still, we wish to do so in order to make our point in the context of a coherent, non-trivial case. The reader should see what is at stake, without first having to look elsewhere. By the same token, he comes to appreciate two facts that are not intuitively obvious and easily glossed over. These are:

1) The number of fitting patterns grows explosively.

2) The best solution may include very poor patterns.

Point 1 is called "the curse of dimensionality" in another context and there it is well illustrated [7, p. 361]. But few people (including the author) respect this curse until it strikes them. They may benefit from seeing that we exceed a computer's memory with as few as 20 different sizes!

As for point 2, the very heart of optimization is to trade a gain here against a loss there, so as to reduce total loss. And in principle an LP procedure succeeds admirably in doing this. In fact it is surprising to see how large a loss you may have to accept on individual rolls.

Awareness of this loss is crucial to the understanding of our heuristic, so we must see exactly how it comes about.

Number of cutting patterns in an absolute procedure

How many patterns must be examined if we want to solve our problem absolutely? The answer cannot be found by a simple formula such as "N square" or "N factorial". This is so because the answer depends not only on the number of sizes at hand, but also on how they happen to fit a given jumbo.

Which data do we examine? The full set in Table 1 produces so many combinations that we cannot hope to list them, but any single column makes a handy set. The data are sorted by width, so each column has a fair share of large and small sizes. You could see each column as an independent set of data, indeed as a day's work.

For example, here are the data in the last column, Friday. They were analyzed by TRIMOPT, our program for the Linear Programming (absolute) answer to the trim problem:

Table 2. Friday only: raw data summary

Sizes	Units	Short	Long	Total	Jumbo	Low
10	116	975	2305	201900	4800	43

Size	No.	Size	No.	Size	No.	Size	No.	Size	No.
2305	4	2135	22	2020	21	1930	10	1830	6
1675	7	1550	20	1360	10	1130	7	975	9

These are the raw data. The summary shows by "Low" how many jumbos are needed if we simply divide total length by jumbo width, here 201900/4800 or 43 units. How many do we need in fact, if we make sure that the reels fit a jumbo?

We must first generate the patterns that fit. This is done by enumerating all possible combinations:

Table 3. Friday: the first 13 patterns for 4800 jumbo

		2305	2135	2020	1930	1830	1675	1550	1360	1130	975
	Loss	-4	-22	-21	-10	-6	-7	-20	-10	-7	-9
#1	190	2	0	0	0	0	0	0	0	0	0
#2	360	1	1	0	0	0	0	0	0	0	0
#3	475	1	0	1	0	0	0	0	0	0	0
#4	565	1	0	0	1	0	0	0	0	0	0
#5	665	1	0	0	0	1	0	0	0	0	0
#6	820	1	0	0	0	0	1	0	0	0	0
#7	945	1	0	0	0	0	0	1	0	0	0
#8	5	1	0	0	0	0	0	0	1	1	0
#9	160	1	0	0	0	0	0	0	1	0	1
#10	235	1	0	0	0	0	0	0	0	2	0
#11	390	1	0	0	0	0	0	0	0	1	1
#12	545	1	0	0	0	0	0	0	0	0	2
#13	530	0	2	0	0	0	0	0	0	0	0

The table heading lists the reels wanted, 2305, 2135, 2020, down to 975; then we have the number wanted: it is marked by a minus sign to signal "demand, as yet unsatisfied". How can we fill the demand for 4 units of 2305, for example? Pattern #1 has 2 units of 2305 on a jumbo of 4800: 2*2305 is 4610, and 4800 - 4610 makes a loss of 190, as shown.

Pattern #2 has only 1 unit of 2305, then 1 of 2135, plus a net loss of 360; similarly, #3 is 2305 plus 2020. We list all combinations starting with 2305, the largest size. Then we turn to those without 2305, but with the next size, 2135.

This process continues down to the bottom of the table:

Table 4. The last patterns for Friday (end of Table 3)

	Loss	2305 -4	2135 -22	2020 -21	1930 -10	1830 -6	1675 -7	1550 -20	1360 -10	1130 -7	975 -9
#84	720	0	0	0	0	0	0	0	3	0	0
#85	950	0	0	0	0	0	0	0	2	1	0
#86	130	0	0	0	0	0	0	0	2	0	2
#87	50	0	0	0	0	0	0	0	1	3	0
#88	205	0	0	0	0	0	0	0	1	2	1
#89	360	0	0	0	0	0	0	0	1	1	2
#90	515	0	0	0	0	0	0	0	1	0	3
#91	280	0	0	0	0	0	0	0	0	4	0
#92	435	0	0	0	0	0	0	0	0	3	1
#93	590	0	0	0	0	0	0	0	0	2	2
#94	745	0	0	0	0	0	0	0	0	1	3
#95	900	0	0	0	0	0	0	0	0	0	4

The last pattern has 4 units of 975. These could be broken down into 3 only (with a loss of 1875), then 2 and finally 1 only. Should we add these extra combinations? That depends on whether you can use a large piece of waste better than an unwanted unit of 975. If both are recycled, you get the same number of jumbos either way, so we can omit the extra rows.

In any case, even omitting such one-size patterns, ten sizes generate as many as 95 combinations. 95 is near 10 squared, so you might think that we have a square function of N, but we shall see below that this is not so.

Poor patterns essential for global optimum

We made it a point to list all possible patterns, good and poor, to clarify the principle. The good patterns certainly look relevant, but must we include also the poor ones?

For example, #95 entails 900 mm of waste, almost as much as the reel of 975 being cut. Can #95 ever beat say #92 which gets 975's at a loss of only 435 mm? Yes, #92 has less loss, but it brings 3 reels of 1130 for each one of 975. So if you need many 975's and few 1130's you prefer pattern #95: it fills total demand with fewer jumbos.

In principle, all patterns must be considered. In fact, not all need to be included initially. Start with only the good ones, those that entail little loss, and see how many jumbos fill demand; then try again with a greater trim margin: if the number of jumbos decreases, increase the trim margin and examine an ever larger number of combinations.

For example, here is a summary of a test run for Friday. We start with the patterns losing at most 100 mm. There are 14 of these, so we might as well list them. To save space, we also show in the column at right the number of patterns to be deckled (optimal global LP solution from TRIMOPT):

Table 5. Best choice from among "good" patterns

Size	2305	2135	2020	1930	1830	1675	1550	1360	1130	975	
Loss	-4	-22	-21	-10	-6	-7	-20	-10	-7	-9	No.
5	1	0	0	0	0	0	0	1	1	0	4
15	0	1	0	0	0	1	0	0	0	1	22
100	0	0	1	0	0	0	1	0	1	0	18
60	0	0	1	0	0	0	0	2	0	0	3
65	0	0	0	1	1	0	0	0	0	1	10
65	0	0	0	1	0	1	0	0	1	0	
10	0	0	0	0	2	0	0	0	1	0	
60	0	0	0	0	1	0	1	1	0	0	
45	0	0	0	0	1	0	0	0	0	3	
90	0	0	0	0	0	2	0	1	0	0	
25	0	0	0	0	0	1	2	0	0	0	1
45	0	0	0	0	0	1	0	0	1	2	
15	0	0	0	0	0	0	1	0	2	1	
50	0	0	0	0	0	0	0	1	3	0	

For example, deckle 4 jumbos for the pattern with a loss of only 5 mm. This yields 4 reels of size 2305, indeed all that is wanted in that size. You also get 4 units each of sizes 1360 and 1130. Demand for these runs to 10 and 7, so some reels still have to come from other patterns.

Where do we get the 22 reels of size 2135? They appear only in pattern 2, the one with a loss of 15 mm. So we obviously need 22 jumbos for that pattern, and that is good enough for 2135. But the same pattern yields also 22 reels of 1675, a size of which we want only 7 units. So 15 are overproduced! 1675 appears again in another inevitable pattern (25 at the bottom of the table), so all in all we get 16 reels surplus.

The best solution formed from these seemingly good patterns finally requires 58 jumbos, 15 more than the "Low" of 43 shown in Table 2. 15 in 43 means a loss of over 1/3!

Why do we use so many jumbos? Because we produce many more reels than we want, 4 of 1830, 16 of 1675, 15 of 1130 and as many as 23 of 975! (Data calculated by hand from Table 5.)

So loss on any one jumbo is at most 100 mm, to be sure. But, having so little choice, we end up with many unwanted reels. Perhaps in taking more loss on some jumbos, we can reduce total usage? Here is a summary for various trim margins:

Table 6. Matrix size and number of jumbos for given trim

Trim	100	300	500	600	640	645	700	900
Rows	14	36	55	66	69	70	72	95
Jumbos	58.00	55.00	53.00	45.67	45.67	44.67	44.67	44.67

The break occurs at 645 mm: the pattern allowing for that much loss is just the one that succeeds in saving a jumbo.

So the good news is that we saved a jumbo and that we need to examine only 70 combinations, instead of 95.

The bad news is threefold:

1) To minimize total loss, we must accept painfully high individual losses, as high as 645 in 4800!

2) By the same token, we are not likely to find the best set of patterns in examining only the good ones (those with little loss per jumbo, say less than 5%).

3) But if we admit poor patterns, we get a number of combinations far beyond an ordinary computer's capacity.

Table size grows explosively

We can handle a table with 50 or 100 patterns/rows such as we had for the 10 different sizes/columns on Friday. But what if we take the initial data set, the one for the whole week, with 54 different sizes? 50 by 50 suggests 2500 rows, but we get many, many more for minimal trim margins:

Table 7. Memory required by various trim margins (K bytes)

Trim	0	5	10	15	20	25	50	100	200
Rows	208	380	572	758	921	1100	2000	3595	6376
Tab	11648	21280	32032	42448	51576	61600	112000	201320	357056
Memory	46	84	126	166	202	241	438	787	1395

The table first lists various trim margins. For example, 0 trim means a perfect pattern without any loss: it is good to see that our data contain so many perfect fits, 208 of them! (If there are so many perfect fits, why do we look further? These fits are perfect in avoiding any loss, but they do not necessarily contain the sizes we want.) Anyway, the perfect patterns alone require a table with as many as 208 rows!

How many columns do we need for the table? One for each size and then two columns to record Pattern Number and Loss, here 54 sizes plus 2 makes 56. So the table as a whole comprises 56*Row cells, for example, 56*208 = 11648, as shown. A cell takes at least 4 bytes (in single precision). This means a total memory requirement of Tab*4/1024 or so many K bytes.

A 640K machine has about 500K free, so we can eventually run about fifty sizes for a trim margin of 50 mm. But this is a far cry from a guarantee of optimality. Nor could we expect to solve a problem of this size in single precision; but if you turn to double precision, you need twice as much space!

So an absolute solution to the trim problem with more than about twenty sizes is impractical on a 640K computer, and a larger machine is impractical on the shop floor.

Compact matrix method?

Our conclusion certainly holds if we store patterns fully, as in Tables 3, 4 or 5. Full storage may mean many zeros, especially in the section of the table holding large sizes. These zeros fix the position of the non-zero entries.

A more elegant approach is to store only non-zero data and to retain their position explicitly in a separate cell: you save the space currently occupied by useless zeros, but two cells are required for each number other than zero. Still, this seems to be a good idea for a table with many zeros.

It may be a good idea for the initial table. But as you go from table to table, the zeros are rapidly replaced by other numbers: each of these now requires two cells! So you run the risk of ending up with higher storage requirements than before (not counting the extra programming load)!

For example, examine the change in density from the first to the last table for our Friday data, those with a trim margin of 100 mm. Table 8 shows at left a section of the initial matrix (copied from Table 5); at right is the corresponding section of the final table for reels that were overproduced:

Table 8. Part of initial and final matrix, "good" patterns

	Initial, from Table 5				Final, after 6 iterations			
Size	1830	1675	1130	975	1830	1675	1130	975
Loss	-6	-7	-7	-9	4.00	16.00	15.00	23.00
5	0	0	1	0	.00	-.25	1.50	.00
15	0	1	0	1	.00	1.00	.00	1.00
100	0	0	1	0	.00	-.50	1.00	.00
60	0	0	0	0	.00	.25	-.50	.00
65	1	0	0	1	1.00	.00	.00	1.00
65	0	1	1	0	-1.00	1.00	1.00	-1.00
10	2	0	1	0	2.00	.00	1.00	.00
60	1	0	0	0	1.00	-.75	.50	.00
45	1	0	0	3	1.00	.00	.00	3.00
90	0	2	0	0	.00	1.75	.50	.00
25	0	1	0	0	.00	.50	.00	.00
45	0	1	1	2	.00	1.00	1.00	2.00
15	0	0	2	1	.00	-.50	2.00	1.00
50	0	0	3	0	.00	-.25	3.50	.00

The data for 1830 and 975 hardly changed, showing that these sizes do not match well with the others. But sizes with many matches have full columns. And they filled up during just 6 iterations. More iterations lead to ever fuller tables.

So it seems that even sophisticated matrix techniques cannot cope with the curse of dimensionality, at least not here.

Successive global optima

If we cannot handle all data at once, how about doing it on a more modest scale, step by step, indeed, day by day?

So find the best solution for each day taken on its own. At the end of the week, add up the number of jumbos used: will it be any worse than the solution for the week as a whole?

Our data set contains enough orders for at least 208 jumbos. We can deckle about 40 jumbos per day, so 208/40 means work for roughly five days. How do we fare if we optimize each day on its own (as if we did not know about the next days)?

Table 9. Raw data grouped by day, from Table 1

Monday Size	Monday No.	Tuesday Size	Tuesday No.	Wednesday Size	Wednesday No.	Thursday Size	Thursday No.	Friday Size	Friday No.
2465	37	2440	6	2385	3	2330	8	2305	4
2235	3	2185	7	2175	10	2160	10	2135	22
2070	31	2035	14	2030	13	2025	5	2020	21
1990	15	1980	16	1955	7	1950	13	1930	10
1920	15	1900	8	1860	23	1850	7	1830	6
1800	9	1780	3	1745	6	1720	10	1675	7
1665	4	1640	8	1600	3	1560	3	1550	20
1525	3	1475	4	1430	2	1380	48	1360	10
1350	7	1235	9	1225	12	1140	9	1130	7
1090	6	1055	2	1015	2	990	5	975	9
965	2	935	2	920	38	845	13		

The data are sorted by size, so Monday has larger units than Friday. But this hardly affects the decision problem. It is to find out, for each day on its own, how many jumbos cover the data available on that day. So do for each day what we did for Friday, find the absolute LP solution via TRIMOPT:

Table 10. Best LP solutions for successive days

Day	Length	Low	Fraction	Integer	Rows	Trim	Setup
Mond	266085	56	59.31	60	81	660	11
Tues	144760	31	32.60	33	104	730	11
Wedn	181580	38	38.68	39	49	240	11
Thur	202980	43	42.90	43	37	140	11
Frid	201900	43	44.67	46	70	645	11
Sum	997305	211	218.16	221	Mean	483	55
Week	997305	208	at most	212	--	200	51

"Length" is the total length of all reels placed end to end. That length divided by jumbo size gives "Low", lowest number of jumbos required. And looking first at the bottom of the summary, we see "Sum", the sum of the daily data and "Week", the result obtained for the week as a unit.

The bottom lines show the same total for the days as for the week, so the data check numerically. But the set taken as a whole needs fewer jumbos than the sum of the daily sets (208 and 211): naturally, the smaller a daily set's total length, the less is it likely to fit an integer multiple of jumbos.

This is shown even more dramatically by the best solutions. We show both the fractional LP and the integer answer, for example, 59.31 and 60 on Monday: on most days going integer costs no more than the fraction rounded up (but it may mean different matches). Only on Friday do the sizes fit poorly: we need 46 jumbos, one more than 44.67 rounded to 45.

The right half of the summary shows how many rows/patterns are included in each LP table. "Trim" is the worst loss accepted on any pattern, running as high as 730 (in 4800). Finally, "Setup" shows the number of patterns in a solution: a change in pattern means changing at least one knife.

So how many jumbos are required to fill all orders? 221 if we operate on a daily basis, disregarding on Monday the data available for the rest of the week. But we use only 212 for the week taken as a whole: 221 - 212 = 9, the number lost by fragmenting the data. A jumbo costs thousands of rands, so the main lesson concerning Global Optimization is clear:

If you must optimize globally, do not fragment your data.

Accumulate orders for as long as you can. Make cutting plans at the last minute, when you have many data.

The global approach can hope to be effective only for a large number of data: only then can you strike a good balance between loss here and gain there.

But the larger the data set, the less can you implement the global approach. This is so for two reasons:

a) Such a set generates so very many cutting patterns that you exceed a computer's capacity, certainly so with an absolute solution procedure such as LP.

b) A large set of data is likely to change while you work on it, so the initial plan becomes obsolete.

A heuristic for global (and local) optimization

We shall show below that you should not even try to optimize globally. It is better to go local. But if you must fill a large set of orders as it stands, what help can we offer?

The help comes from seeing the essential idea of the global approach in an operational perspective: what does it do? The idea is to accept a loss here for a gain there (as long as you reduce total loss). You would naturally understand this to mean a small loss and a big gain. This is not so.

The critical point is this: the loss to be accepted in one pattern may be quite large, so large that a deckler refuses to consider it, like 645 or even 730 in 4800: he would never let such a piece go to waste. (The deckler may even refuse your computer plans and work to his own designs if you let him, the more so if he does not see why a large loss here should bring a greater gain there).

But now that we understand this fact, we might as well bow to it gracefully. In other words, if you must work globally, do not try to do too well on any one jumbo (by setting your trim margin to zero or some other low number like 5 or 10): in striving for perfection on one jumbo you run the risk of missing the lot! Rather, set the trim margin to 200, 300 or even higher, then accept any pattern respecting that margin.

This is the basis of our heuristic procedure. It does not use any mathematical/LP-type formula, but scans the data for matching partners. Its critical feature is this: the trim margin is not a mere statistic, to be minimized; rather, it decides whether a match is accepted

So the margin itself is part of the analysis. Specifically, set a trim margin and then --

1) Organize your data by some order of priority (below), then assume that the first reel must be cut, now.

2) Scan the remaining reels for suitable partners: accept the first match that respects the current trim margin.

   If no partner is found within trim margin, accept the next best alternative (or go local, see below).

3) Update Number of Jumbos and List of Reels, then repeat Step 1 until the list is empty.

4) Change the trim margin and repeat procedure; continue doing so until you find the lowest number of jumbos.

How good is this procedure? The critical data appear in the columns LP and TRIM: it is the number of jumbos used to fill the orders for each day, then for the week as a whole:

Table 11. Best LP solutions compared to heuristic TRIM

Day	Length	Low	Loss	Setup	LP	TRIM	Margin	Setup
Mond	266085	56	660	11	60	60	345	11
Tues	144760	31	730	11	33	33	785	7
Wedn	181580	38	240	11	39	39	105	12
Thur	202980	43	140	11	43	43	140	10
Frid	201900	43	645	11	46	46	0	13
Sum	997305	211	483	55	221	221	--	53
Week	997305	208	Mean	--	--	212	200	51

The evidence is overwhelming: we never do worse than the absolute LP procedure, certainly not on any given day. For the week as a whole (with all 54 sizes at once), we do not know the true absolute answer, but the heuristic answer of 212 is so good that we are hardly motivated to look further.

"Loss" is the largest trim lost by a global solution (from TRIMOPT). Conversely, "Margin" is the best trim margin for Step 2, the limit for accepting a pattern, even a poor one: in going global, you must eventually tolerate large losses. Anyway, the best margin reduces jumbo usage to the lowest level (other margins may use many more jumbos). "Setup" is the number of changes from one pattern to the next one.

You may want more empirical evidence for our conclusion, but you have seen five experiments already. Our data come from one set, to be sure, but we split it into five columns and a column is as good as a set of its own. And if five sets do not convince you, ten won't either.

In any case, the critical evidence comes from the mind, not from data. It is to understand the crucial aspect of global optimization. This is to tolerate a high loss on individual jumbos if that reduces the total number of jumbos.

We apply the same reasoning. We may expect similar results.

Priority used in organizing data

Step 1 is to arrange reel sizes according to some priority.

The most likely criterion is date of delivery and we provide for orders to be sorted and deckled accordingly. Also, as he goes along, the deckler may declare any order to be urgent: it moves to the front and is done first, regardless of loss, but so as to fill most of that order at once. Declaring all orders to be "urgent" tends to reduce the number of setups.

How many jumbos do we need if we respect delivery dates? One way to simulate that question is to run our "daily" data (by column rather than by row): the data for Monday get priority over those for Friday, but we do not prevent a Friday reel from joining one for Monday as a second or third piece.

Respecting delivery dates in this way means 217 jumbos. This compares to 212 for the full week's data and to 221 for five successive independent daily solutions (finish Monday before you even look at Tuesday). See also "local" example below.

If no orders are urgent (or if all are equally urgent), we must think of some other criterion of priority. The natural answer is to go by Difficulty of Cut: start with the more difficult pieces. But how do we measure difficulty? By --

a) Length of a reel: long reels are more difficult than short ones, so arrange the data in descending order.

b) Loss if reel is placed on its own: 4800/1600 is clean, but 4800/1500 is not. So start with the worst reel.

Policy (a) is more effective; (b) pulls too many short reels to the front. So individual difficulty hardly counts. This is as it should be: a size is never difficult in itself; the difficulty lies in matching it with partners, and these are hard to find among few data.

A different criterion is magnitude of order: consider reel size as well as number wanted, to favor the large order. But a simple multiple of size*number again pulls too many short reels to the front. What if we weight reels by their size, then multiply? Try size*size*number to decide priority.

Table 12. Best TRIM results for different priorities

First reel set by			Size*Size*Number			Simple Size		
Day	Length	Low	TRIM	Margin	Setup	TRIM	Margin	Setup
Mond	266085	56	60	345	11	60	345	12
Tues	144760	31	33	785	7	33	785	8
Wedn	181580	38	39	105	12	39	320	11
Thur	202980	43	43	140	10	43	140	10
Frid	201900	43	46	0	13	46	645	10
Sum	997305	211	221	255	53	221	447	51
Week	997305	208	212	200	51	214	595	51

On the short daily runs both methods perform equally well, though squaring the size leads to lower trim margins. On the long run it saves two jumbos, 212 compared to 214.

Truly local optimization

Our procedure meets the critical requirement of the global approach: it is to fill orders as they stand, now, without any chance of revision. This requirement implies Step 1,

1) ... assume that the first reel must be cut, now.

If all reels must go as they stand, some must eventually be cut regardless of loss. Our only chance to reduce loss is to cut first the reel that is most in need of good partners. And by choosing a good trim margin, we may expect to perform as well as an absolute procedure (but work faster and within a modest memory space, indeed on a machine with 128K).

But think again. Why should the first (or any reel) have to be done now, indeed today instead of tomorrow or next week? There are deadlines, to be sure, but chances are that they are as bad next week as they are now. Also, we can do urgent items with priority. Anyway, a set of orders as in our basic example takes several days to be deckled: so why not leave the bad pieces for the end of the run, or until new orders have arrived? Deal now with the good ones only!

What happens if you do? How many good matches are there? Our example contains 208 perfect patterns, but of course not all of these are useful (some contain reels rarely sold). So run our program with a view to answering this question:

Using the orders in hand, how many jumbos can be filled without loss on one jumbo ever exceeding a given margin?

Here are the answers for margins up to 200 mm:

Table 13. Number of jumbos with loss within trim margin

Trim	0	5	10	20	30	50	100	200
Jumbos	117	142	149	144	153	151	154	196

117 out of 208 jumbos, more than half of all orders, can be deckled without any loss at all! We could be busy for three days in a row without losing one millimeter in 576000! This may seem too good to be true, but it is true. It is equally astounding that we get 149 jumbos out of 208 with at most 10 mm lost. To be exact, total loss comes to 590 mm in 715200, less than 4 mm per jumbo! So roughly speaking,

2/3 if not 3/4 of all orders go with hardly any loss!

Most of the loss comes from the remaining 1/3 of orders.

If we can deckle so many reels without loss, do we get those wanted for Monday? Not necessarily. Our example assumed that the data were sorted by size, not by date. But you can also go by date: put the data for Monday first, but do use reels for subsequent days if they fit better than those for Monday on its own. How many jumbos must be deckled before Monday's demand is filled? 104 jumbos in all, with a total loss of 990 mm, less than 10 mm per jumbo! 104 jumbos are more than you can do on Monday alone, so start on Friday.

We said "most of the loss comes from 1/3 of orders". These are the reels left at the end of the run. Why do they cause so much loss? Not because they are poor sizes, but because no pieces are left to match them. We noted this previously when discussing "difficult" sizes: no size is inherently difficult; it becomes so only through lack of partners. Partners tend to lack whenever you have few data.

To see that this is so, run the full set of 507 reels and 54 different sizes and deckle 149 jumbos with a trim margin of at most 10 mm. After doing that, you still need --

Table 14. Reels left after four days "local" work

Sizes	Reels	Min	Max	Total	Jumbo	Low
21	138	1550	2465	282695	4800	59

Size	No.	Size	No.	Size	No.	Size	No.	Size	No.
2465	13	2440	6	2385	3	2305	4	2235	3
2185	6	2175	6	2160	10	2135	6	2070	10
2030	4	1990	8	1980	8	1950	11	1920	8
1860	9	1850	2	1830	4	1780	3	1720	1
1550	13								

There are small sizes and large ones, but none is inherently more unsuited for jumbos of 4800 than the reels already cut. It is just that those reels had others to match and that the matching sizes are now used up. So await orders for more of the missing sizes, before you deckle those left now.

What if you must deckle the remaining reels now, before new orders arrive? If so, you should run the initial lot with a margin of 200 mm, not first with a low margin, then with a margin yet to be found: this keeps usage to 212 jumbos.

But if you insist on separate sets, run the remaining reels globally (or use TRIMOPT): you need another 67 jumbos. So 149 near perfect jumbos plus 67 poor ones makes 216 in all: 216 - 212 or 4 jumbos are lost because we divided the data into two sets, good matches and poor matches (but we lost 9 jumbos when we divided the data into five daily sets).

Losing four jumbos in 212 is the possible Cost of Perfection -- on most of your work. The cost is due to the risk of not getting the new orders that would make perfection of the bad sizes, too. The risk should be negligible in a company with a continuous stream of orders.

Implementation

Data are required not only for size and number of reels, but also for paper type, delivery, client information and so on. So data input is naturally done in the office (via TRIMRAW, a program designed for data capture and file creation). Once the data are on disk, the disk can be used in the factory: our program can run there and then, on an ordinary desk-top computer. The operator himself can print his deckling plans as he goes along, certainly so if he optimizes locally.

In fact, at Mondi deckling plans continue to be made in the office, by the former master deckler: he enjoys running our program much more than he did his attempts at matching reels paper and pencil in hand.

So the deckler has not lost his job. On the contrary, now he works effectively: he no longer juggles numbers, but he organizes cutting plans so as to integrate delivery dates with production schedules. He also decides whether a set of orders can be run locally or whether he must eventually get an absolute, global solution (if deadlines are to be met). If so, he finds the best global trim margin via TRIMARG.

References

[1] Paul, A. E. and J. R. Walter, "The Trim Problem: An Application of Linear Programming to the Manufacture of Newsprint Paper", paper presented at the Annual Meeting of the Econometrica Society, Montreal, September 1954. Published 1956 in Pulp Paper Magazine of Canada, Vol. 57.

Other early references are: K. Eisemann, "The Trim Problem", Management Science, 3 (3) 279-284 (April 1957) and S. Vajda, "Trim Loss Reduction" in Readings in Linear Programming. New York: John Wiley, 1958, pp. 78 - 84.

[2] Cook, Thomas M. and Robert A. Russel, Introduction to Management Science. Englewood Cliffs: Prentice-Hall, 1981, pp. 61 - 62.

[3] Haessler, R. W., "Developing an industrial-grade heuristic problem solving procedure", Interfaces, 13 (3) 62-71 (1983)

[4] Hinxman, A. I., "The trim-loss and assortment problems: A survey", European Journal of Operational Research, 5 (1) 8-18 (1980)

[5] Derek McAslan, Paper and Board Mill Process Engineer at Mondi Paper Company, P. O. Box 1551, Richards Bay 3900.

[6] Schrage, Linus E. Linear Programming Models with LINDO. Palo Alto: The Scientific Press, 1981, pp. 66 - 70.

[7] Taha, Hamdy A. Operations Research: An Introduction. New York: Macmillan, 1982, pp. 34 - 37.