ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1996
Seria: AUTOMATYKA z. 118 Nr kol. 1338

Krzysztof PIEŃKOSZ
Politechnika Warszawska

KOMPUTEROWY SYSTEM OPTYMALIZACJI ROZKROJU SZKLANYCH TAFLI

Streszczenie. W pracy przedstawiono system komputerowy oraz zastosowany algorytm do optymalizacji rozkroju szklanych tafli. Zadaniem programu jest rozplanowanie rozkroju tafli na prostokątne szyby o zadanych wymiarach w taki sposób, aby były minimalizowane odpady. Uwzględniane są przy tym różnorodne wymagania technologiczno-organizacyjne ograniczające dopuszczalne sposoby cięcia.

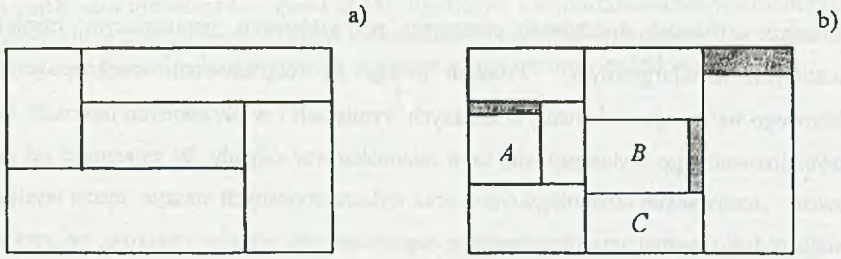
A COMPUTER SYSTEM FOR GLASS PLATE CUTTING OPTIMIZATION

Summary. In the paper a computer system and the applied algorithm for glass plate cutting optimization is presented. The aim of the program is to plan the cutting of glass plates into rectangular pieces of given dimensions as to minimize the waste. Various production and organization demands which restrict the feasible cut patterns are taken into account.

1. Wprowadzenie

Zagadnienie optymalizacji rozkroju materiałów pojawia się jako element planowania produkcji w różnych dziedzinach przemysłu, np. szklarskim, papierniczym, meblarskim, odzieżowym, metalurgicznym. Problem polega na rozplanowaniu rozkroju materiału wyjściowego na mniejsze elementy o zadanych wymiarach i w określonych ilościach, tak aby maksymalizować jego wykorzystanie, czyli minimalizować odpady. W zależności od rodzaju surowca, stosowanych technologii cięcia oraz wykorzystywanych maszyn mogą występować różnego rodzaju wymagania ograniczające dopuszczalność wzorów rozkroju, co sprawia, że programy wspomagające rozkrój muszą być w zasadzie specyficzne dla każdej branży, zakładu czy maszyny.

Przedstawiany w pracy system został zaprojektowany na zamówienie Zakładu Szyb Zespolonych wytwarzającego elementy do okien. Wszystkie szyby są cięte z tafli o takich samych wymiarach 3210 mm na 2250 mm. Występują cztery rodzaje (grubości) szkła. Zamówienia na szyby mają charakter jednostkowy, tzn. wymiary zamawianych szyb są z reguły w każdym zleceniu inne. Powtarzalność tych samych wymiarów szyb w obrębie jednego zlecenia też jest przeważnie niewielka. Szyby są prostokątne, w związku z tym wykonywane są tylko cięcia poziome i pionowe. Cięcia muszą mieć charakter gilotynowy, tzn. muszą przebiegać od brzegu do brzegu tafli lub wcześniej odciętej jej części. Niemożliwe więc jest np. wykrojenie szyb według wzoru przedstawionego na rys. 1a. Używana maszyna tnąca umożliwia wykonanie tylko trzech stopni cięć – w pierwszej fazie tnie się taflę na pionowe pasy, w drugiej fazie są wykonywane cięcia poziome i w trzeciej fazie ewentualnie jeszcze raz pionowe. Wobec takich ograniczeń niemożliwe jest np. zrealizowanie rozkroju według wzoru na rys. 1b. Wycięcie szyby oznaczonej na rysunku symbolem *A* wymaga bowiem wykonania czterech stopni cięć. Szerokość pasów ciętych w fazie pierwszej i trzeciej nie może być mniejsza niż 100 mm, a w fazie drugiej 300 mm. Po wykonaniu rozkroju szyby są łamane wzdłuż linii cięć w celu ich rozłączenia. Z przyczyn technologicznych odłamywane kawałki odpadów nie mogą być węższe od zadanej progowej szerokości zależnej od grubości szkła. Z tego powodu, jeżeli na rys. 1b szyba *B* jest krótsza jest od szyby *C*, to musi ona być krótsza co najmniej o zadany margines, ewentualnie cięcie musi przebiegać nieco dalej na prawo, tak aby zarówno dla szyby *B* jak i *C* żądany margines był zachowany. Marginesy technologiczne są także wymagane na brzegach tafli.

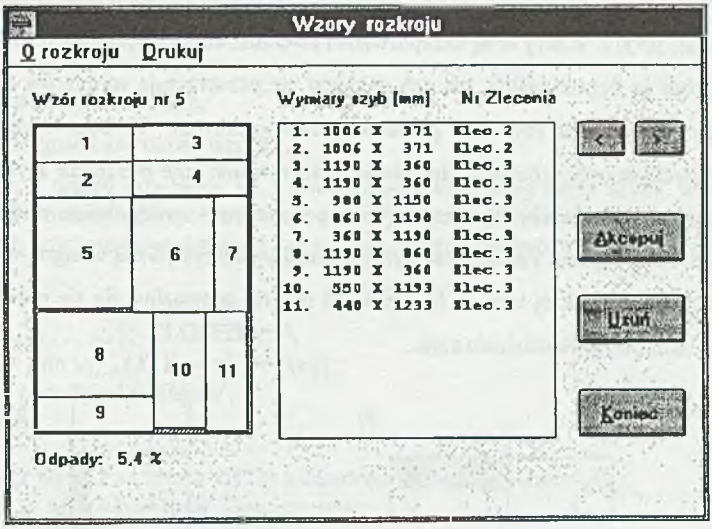


Rys. 1. Przykłady niedopuszczalnych wzorów rozkroju
Fig. 1. Examples of infeasible cutting patterns

2. Opis systemu

Spełnienie wymagań przedstawionych w poprzednim punkcie wymagało stworzenia specjalizowanego oprogramowania. Funkcje systemu zostały podzielone na trzy niezależne moduły: moduł obsługi zleceń, moduł wyznaczania wzorów rozkroju i moduł prezentacji.

Moduł obsługi zleceń służy do rejestracji wpływających zamówień, przeglądania zawartości zleceń i stanu ich wykonania oraz usuwania z pamięci zleceń zrealizowanych. Moduł wyznaczania wzorów rozkroju umożliwia przeglądanie zleceń oczekujących na realizację, wybór zleceń do rozkroju oraz uruchomienie procedury wyznaczania wzorów rozkroju. Moduł prezentacji jest wywoływany automatycznie po wyznaczeniu wzorów rozkroju. Można go też wywołać niezależnie z głównego poziomu programu. Umożliwia przegląd utworzonych wzorów rozkroju pod kątem analizy ich struktury oraz generowanych odpadów. Wygląd ekranu wyświetlanego przez ten moduł dla przykładowego rozwiązania jest przedstawiony na rys. 2. Poszczególne wzory rozkroju można przejrzeć, odrzucić i usunąć z pamięci lub zaakceptować, co jest odnotowywane w bazie zleceń jako fakt realizacji szyb występujących we wzorze. W każdym momencie można wydrukować aktualny zestaw wzorów oraz uzyskać informacje statystyczne dotyczące rozkroju. System pracuje w środowisku MS Windows i sposób jego obsługi jest zgodny z tym standardem.



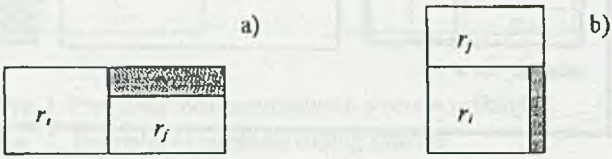
Rys. 2. Wyświetlanie wzorów rozkroju

Fig. 2. The cutting patterns display

3. Algorytm optymalizacji rozkroju

Problem optymalizacji rozkroju jest złożonym zadaniem kombinatorycznym. W literaturze można znaleźć wiele różnych podejść do rozwiązywania tego problemu opartych na programowaniu liniowym [6,8], programowaniu dynamicznym [2], metodzie podziału i oszacowań [3,4], sklejaniu wzorów rozkroju [5,9,10], generowaniu i upakowywaniu pasów szyb [1,11]. Większość tych metod ma charakter heurystyczny i sprowadza się do rozplanowywania poszczególnych tafli po kolei, tzn. najpierw pierwszej tafli, potem drugiej itd. Jedynie modele programowania liniowego ujmują problem w sposób globalny, ale ich użyteczność ogranicza się do zadań o niewielkim zróżnicowaniu wymiarów szyb. Ponadto trudno jest tu uwzględnić specyfikę ograniczeń technologicznych, np. wymagania na marginesy odcięć.

W prezentowanym w pracy systemie wzory rozkroju są wyznaczane niezależnie dla każdej tafli z osobna. Do rozkroju pojedynczej tafli zastosowano algorytm bazujący na metodzie Wanga [10]. Proces tworzenia wzoru rozkroju odbywa się „od dołu" poprzez sklejanie mniejszych wzorów aż do wypełnienia całej tafli. Wzorami wyjściowymi, od których rozpoczyna się sklejanie, są pojedyncze szyby. W danym kroku algorytmu wybiera się dwa cząstkowe wzory rozkroju i tworzy się z nich dwa następne poprzez sklejenie poziome oraz pionowe (patrz rys. 3). W rezultacie, z dwóch wzorów cząstkowych r_i oraz r_j o wymiarach (x_i, y_i) oraz (x_j, y_j) otrzymujemy dwa nowe wzory o wymiarach $(x_i+x_j, \max\{y_i, y_j\})$ oraz $(\max\{x_i, x_j\}, y_i+y_j)$. Wzory te są akceptowane i ewentualnie uwzględniane w procesie dalszych sklejeń, jeżeli są dopuszczalne, tzn. ich wymiary nie przekraczają wymiarów tafli oraz liczba występujących w nich szyb nie przekracza zapotrzebowań. W celu ograniczenia liczby generowanych wzorów rozkroju stosowane są tu również inne ostrzejsze kryteria akceptacji, zapobiegające powtarzaniu się wzorów oraz odrzucające wzory, które według oszacowań dadzą zbyt duże odpady. Podstawową właściwością wykorzystywaną w algorytmie jest fakt, że każdy gilotynowy rozkrój tafli, w tym również rozkrój optymalny, da się odtworzyć poprzez poziome i/lub pionowe sklejenia szyb.



Rys. 3. Sklejenia wzorów: a) poziome, b) pionowe
Fig. 3. Builds of patterns: a) horizontal, b) vertical

Proces przeglądania i generacji wzorów bazuje na procedurze *Best-First Search* zaczerpniętej z metodologii przeszukiwań stosowanych w metodach sztucznej inteligencji [5,7,9]. Rozpatruje ona w pierwszej kolejności te rozwiązania (wzory cząstkowe), które mają najmniejsze wartości oszacowań spodziewanych odpadów. Dla każdego wzoru r niech $g(r)$ oznacza *odpad wewnętrzny*, czyli wielkość tej części powierzchni wzoru, która pozostanie po wykrojeniu z niego szyb. Z kolei niech $h(r)$ będzie oszacowaniem od dołu *odpadu zewnętrznego*, który powstanie na zewnątrz wzoru r, jeżeli zostanie on umieszczony na tafli. Oczywiście, dokładna wielkość odpadu zewnętrznego będzie zależeć od wymiarów i ilości szyb, które znajdą się na tafli obok wzoru r. Rozmiar całkowitego odpadu związanego z rozkrojem wzoru r można oszacować od dołu jako $f(r)=g(r)+h(r)$. Załóżmy, że funkcja f jest tak dobrana, że jeżeli wzór r_k powstał w wyniku sklejenia wzorów r_i oraz r_j, to

$$f(r_k) \geq \max\{ f(r_i), f(r_j) \}. \tag{1}$$

Zauważmy, że taka właściwość zachodzi, gdy np. przyjmiemy oszacowanie odpadu zewnętrznego równe $h(r)=0$ dla każdego r. Wówczas $f(r_k)=g(r_k)\geq g(r_i)+g(r_j)=f(r_i)+f(r_j) \geq \max\{f(r_i), f(r_j)\}$. W praktyce, w celu zwiększenia efektywności algorytmu, dąży się jednak do uzyskania bardziej dokładnych oszacowań odpadu zewnętrznego. Poniżej przedstawiono ogólny schemat zastosowanego algorytmu.

Algorytm wyznaczania wzorów rozkroju

Oznaczenia:

X, Y – długość i szerokość tafli,

$x(r)$, $y(r)$ – długość i szerokość wzoru r,

$r_1, r_2, \ldots, r_n$ – wzory składające się z pojedynczych szyb, przy czym szyby, które nie są kwadratowe, muszą być reprezentowane podwójnie (z zamienionymi wymiarami długości i szerokości), gdyż mogą występować we wzorze rozkroju w dwóch różnych orientacjach.

begin
  OPEN:={ $r_1, r_2, \ldots, r_n$ }; CLOSED:= ∅;
  MinOdpad:= min $_{i=1,\ldots,n}\{X*Y - x(r_i) * y(r_i)\}$;
  r^*:= wzór o odpadzie *MinOdpad* ;
  Koniec:=**false;**
    repeat
      wybierz wzór r ze zbioru OPEN z najmniejszym oszacowaniem $f(r)$;
      if $f(r) \geq$ *MinOdpad* **then** *Koniec*:= **true;**
      else
        przesuń r ze zbioru OPEN do zbioru CLOSED;
        for każdego q będącego sklejeniem poziomym lub pionowym
          wzoru r i wzoru ze zbioru CLOSED **do**

```
            if wzór q dopuszczalny and f(q) < MinOdpad then
                MinOdpad:=min{X*Y – x(q)*y(q) + g(q), MinOdpad} ;
                r*:= wzór o odpadzie MinOdpad ;
                dołącz q do zbioru OPEN ;
    until Koniec or OPEN=∅;
    wyświetl wzór r* ;
end
```

W trakcie działania algorytmu wzory rozkrojów są przechowywane w dwóch zbiorach OPEN i CLOSED. Zauważmy, że jeżeli $q \in$OPEN i $r \in$CLOSED, to $f(q) \geq f(r)$. Do zbioru CLOSED są bowiem przenoszone zawsze te wzory r ze zbioru OPEN, które mają najmniejsze wartości oszacowań $f(r)$. Zgodnie z zależnością (1), dla nowo generowanych wzorów q umieszczanych w zbiorze OPEN, wartość $f(q)$ nie może być mniejsza od $f(r)$. Zmienna *MinOdpad* przechowuje na bieżąco informacje o wzorze, który w danej chwili ma najmniejszy odpad. Gdy w algorytmie zostanie spełniony warunek $f(r) \geq MinOdpad$, to mamy gwarancję, że w zbiorach OPEN i CLOSED nie ma już lepszych wzorów niż wzór r^* o odpadzie *MinOdpad*. Zgodnie z zależnością (1), w wyniku dalszego sklejania też nie uzyska się już lepszych wzorów. Algorytm wyznacza więc rozkrój tafli w sposób optymalny. Należy jednak wyraźnie zaznaczyć, że w przypadku rozkroju sekwencji kilku tafli i stosowania algorytmu dla każdej tafli niezależnie, uzyskane rozwiązania nie muszą być globalnie optymalne.

Sprawdzenie dopuszczalności generowanego wzoru rozkroju q polega na zbadaniu, czy jego wymiary nie przekraczają wymiarów tafli – z uwzględnieniem marginesów technologicznych. Ponadto, liczba szyb występująca we wzorze nie może być większa od liczby zamawianej. Ze względu na ograniczenia technologiczne nie wszystkie rodzaje sklejeń są też dopuszczalne (patrz rys. 1). Dodatkowo sprawdza się, czy $f(q) < MinOdpad$, gdyż w przeciwnym przypadku zastosowanie wzoru q na pewno nie da mniejszego odpadu od odpadu wzoru r^*.

4. Implementacja algorytmu

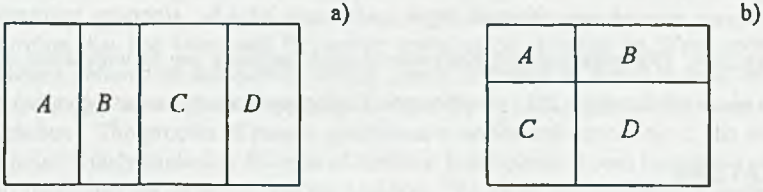
Głównym problemem przy implementacji algorytmu jest przechowywanie i przeglądanie generowanych wzorów rozkroju. Dlatego dużo uwagi poświęcono na ograniczenie ich liczby i zastosowanie odpowiednich struktur danych. O efektywności algorytmu w dużej mierze decyduje jakość oszacowań. Gdybyśmy dysponowali dokładnymi oszacowaniami odpadu zewnętrznego h, rozwiązanie znajdowane by było bardzo szybko, generując przy tym

niewielką liczbę wzorów cząstkowych. W praktyce uzyskanie dobrych oszacowań, o ile w ogóle możliwe, jest skomplikowane i czasochłonne. Z konieczności stosowane są oszacowania w miarę łatwe do wyliczenia, ale dość zgrubne. Oparte są one na analizie potencjalnie dopuszczalnych szerokości i długości wzorów. W rezultacie liczba generowanych cząstkowych wzorów rozkroju gwałtownie rośnie.

W procesie sklejania pewne wzory mogą się powtarzać i niepotrzebnie zajmować miejsce w pamięci oraz generować następne powtórzenia. Zauważmy, że np. wzory przedstawione na rys. 4 można otrzymać w wyniku różnej sekwencji sklejeń. Wzór na rys. 4a powstaje zarówno jako efekt sklejenia $((A + B)+C)+D$, jak i $(A+B)+(C+D)$, czy też $A+(B+(C+D))$, gdzie znak „+" symbolizuje sklejenie, a nawiasy określają kolejność ich wykonywania. Co więcej, nie jest też ważne, jak szyby A,B,C,D są ułożone względem siebie, o ile zewnętrzne wymiary całego wzoru są zachowane. Np. sklejenia $((A+D)+C)+B$ utworzą wzór równoważny, który też stanowi pewnego rodzaju powtórzenie. Na rysunku rys. 4b pokazano inny przykład powtórzenia, gdzie wzór może powstać na dwa różne sposoby poprzez sklejenia $(A+B)+(C+D)$ lub $(A+C)+(B+D)$.

Aby uniknąć powtórzeń, zastosowano następujące reguły sklejania wzorów:

- do wzoru q będącego sklejeniem poziomym $r_i + r_j$ nie może być doklejany poziomo wzór r_d, który jest sklejeniem poziomym innych wzorów. Wyklucza to sekwencje sklejeń poziomych typu $(A+B)+(C+D)$, które dają taki sam efekt jak $((A+B)+C)+D$. Dodatkowo wymaga się, aby doklejany wzór r_d miał szerokość nie większą niż szerokości wzorów r_i oraz r_j. Ustala to jednoznacznie kolejność doklejania wzorów według nierosnących szerokości,
- tak jak wyżej, tylko dla przypadku sklejeń pionowych,
- do wzoru q, będącego sklejeniem poziomym $r_i + r_j$, nie może być doklejany pionowo wzór, którego szerokość lub szerokość jednego z jego wzorów cząstkowych jest równa szerokości wzoru r_i lub r_j. Wyklucza to sklejenia typu $(A+B) + (C+D)$ z rys. 4b na rzecz sekwencji $(A+C) + (B+D)$.



Rys. 4. Przykłady potencjalnych powtórzeń wzorów rozkroju
Fig. 4. Examples of possible duplications of cutting patterns

Dalsze zmniejszenie liczby generowanych wzorów rozkroju uzyskano odrzucając, podobnie jak w [10], wzory o zbyt dużej powierzchni odpadu wewnętrznego – przekraczającej pewien ustalony próg. W zależności od wariantu algorytmu, wzór q jest odrzucany, jeżeli:

$$g(q) > \beta_1 * X * Y \quad \text{lub} \tag{2}$$

$$g(q) > \beta_2 * x(q) * y(q) \tag{3}$$

gdzie β_1, β_2 są parametrami progowymi z przedziału [0,1]. Stosowanie reguł (1) i (2) jest w ogólnym przypadku działaniem restrykcyjnym i może spowodować odrzucenie przez algorytm wzorów prowadzących do rozwiązań optymalnych. W przypadku stosowania reguły (2) okazuje się jednak, że jeżeli po zakończeniu algorytmu jest spełniony warunek

$$\beta_1 \geq MinOdpad / (X * Y) \tag{4}$$

to wzór końcowy r^* o powierzchni odpadu równej *MinOdpad* jest optymalny. Wynika to z faktu, że skoro zachodzi (4), to w trakcie działania algorytmu reguła (2) mogła odrzucić tylko wzory o odpadzie większym niż *MinOdpad*. W praktyce stosowana jest raczej reguła (3), która jest zwykle bardziej restrykcyjna i odrzuca więcej wzorów cząstkowych, ograniczając przez to ich liczbę do rozsądnych rozmiarów. Doświadczenia pokazują, że przy odpowiednio dobranym parametrze β_2 niewiele się traci na jakości rozwiązań w stosunku do reguły (2), za to dużo zyskuje na szybkości obliczeń i zajętości pamięci.

5. Uwagi końcowe

Przedstawiony system optymalizacji rozkroju został wdrożony w Zakładzie Szyb Zespolonych w Lublinie. Poprzednio prace związane z wyznaczaniem wzorów rozkroju były wykonywane ręcznie. Zastosowanie komputera zdecydowanie usprawniło proces rozkroju, umożliwiając przyśpieszenie przygotowywania wzorów, lepsze wykorzystanie pracowników i uzyskanie znaczących oszczędności w zużyciu szkła.

Zebrane doświadczenia wskazują, że zastosowany algorytm jest bardzo elastyczny. Pozwala uwzględnić szereg praktycznych ograniczeń wynikających z wymagań technologicznych. Mimo użycia stosunkowo prostych reguł szacowania odpadu, algorytm wykazuje się dużą efektywnością. Dla praktycznych zestawów danych uzyskuje się w większości przypadków odpady nie przekraczające 5% , po odliczeniu niezbędnych marginesów technologicznych.

LITERATURA

1. Albano A., Orsini R.: A heuristic solution of the rectangular cutting stock problem. The Computer Journal, vol. 23, 1980, pp. 338-343.

2. Beasley J.E.: Algorithms for unconstrained two-dimensional guillotine cutting. J. Opl. Res. Soc., vol. 36, 1985, pp. 297-306.
3. Christofides N., Whitlock Ch.: An algorithm for two-dimensional cutting problems. Operations Research, vol. 25, 1977, pp. 30-44.
4. Christofides N., Hadjiconstantinou E.: An exact algorithm for orthogonal 2-D cutting problems using guillotine cuts. European Journal of Operational Research, vol. 83, 1995, pp. 21-38.
5. Daza V.P., de Alvarenga A.G., de Diego J.: Exact solutions for constrained two-dimensional cutting problems. European Journal of Operational Research, vol. 84, 1995, pp. 633-644.
6. Gilmore P., Gomory R.E.: A linear programming approach to the cutting stock problem - Part II. Operations Research, vol. 11, 1963, pp. 863-888.
7. Pearl J.: Heuristics, intelligent search strategies for computer problem solving. Addison-Wesley Publishing Company Inc., 1984.
8. Piasecki B.: Optymalny rozkrój materiałów. WNT, Warszawa 1978.
9. Viswanathan K.V., Bagchi A.: Best-first search methods for constrained two-dimensional cutting stock problems. Operations Research, vol. 41, 1993, pp. 768-776.
10. Wang P.Y.: Two algorithms for constrained two-dimensional cutting stock problems. Operations Research, vol. 31, 1983, pp. 573-586.
11. Zissimpopoulos V., Fayard D.: An approximation algorithm for solving unconstrained two-dimensional knapsack problems. European Journal of Operational Research, vol. 84, 1995, pp. 618-632.

Recenzent: Dr hab.inż. Konrad Wala. prof. AGH

Wpłynęło do Redakcji do 30.06.1996 r.

Abstract

The combinatorial optimization problem considered in the paper is a special two dimensional cutting stock problem arising in the glass industry. Given a demand of non oriented small rectangular pieces, the problem is to determine how to cut them out of the large plates as to minimize the waste. There are some specific constraints on the cuts feasibility following from organization and production restrictions. All cuts must be orthogonal, i.e. parallel to one side of the plate. Moreover, any cut must be of guillotine type, i.e. it must run from edge to edge of the rectangle being cut. Only three stages of cuts are permitted. A computer program to support cutting planning is presented. It runs under MS Windows control and enables to register coming demands, generates cutting patterns and provides displaying and printing functions.

The algorithm that has been used for cutting optimization is based on Wang method. It generates cutting patterns by successively adding pieces or groups of pieces to each other and building up larger patterns in bottom-up way. The pattern that has the smallest waste is taken to be the solution. The process of pattern generation is performed according to the *best-first search* technique widely studied in the area of Artificial Intelligence. It uses trim loss evaluation function to determine the order of pattern building. To avoid the explosive growth of the number of partial solutions being generated, duplicate pattern constructions are rejected and restrictions on acceptable level of internal trim losses are imposed.