AUTOMATYKA • 2010 • Tom 14 • Zeszyt 3/2

Bogusław Filipowicz*, Wojciech Chmiel*, Piotr Kadłuczka*, Konrad Wala**

Kontur wypukły w trójwymiarowym zagadnieniu pakowania

1. Wprowadzenie

Magazynowane w hurtowni towary są dystrybuowane do klientów zgodnie z uzgodnionym zapotrzebowaniem. Klientami hurtowni są zwykle inne hurtownie, firmy produkcyjne, handlowe i usługowe. Kompletacja i pakowania wysyłek jest ważnym składnikiem technologicznego procesu magazynowego i spedycyjnego. Sposób realizacji tego etapu decyduje w znacznym stopniu o efektywności procesów logistycznych wewnątrz przedsiębiorstwa, jak również o efektywnej współpracy z partnerami handlowymi w ramach logistycznych łańcuchów dostaw. Rozwiązania systemów kompletacyjnych muszą uwzględniać aktualne wymagania rynku dotyczące: terminowości (ścisłego harmonogramowania) dostaw, jakości dostarczanych towarów, zgodności przesyłek z zamówieniami, znakowania w sposób umożliwiający ich automatyczną identyfikację wewnątrz magazynu oraz u partnerów handlowych, a także zapewniać realizację zasady *Just-in-Time* i rozwiązań oszczędnych *Lean*, warunkujących optymalizację kosztów. W celu uzyskania zadawalającej wydajności zautomatyzowanych systemów kompletacji i pakowania stosowane są programy sterujące przepływem ładunków, dyspozycji i dokumentów, gdzie szczególnego znaczenia nabierają efektywne metody i algorytmy rozwiązywania złożonych problemów optymalizacji dyskretnej.

Innowacyjność w systemach logistycznych ma decydujący wpływ na efektywność przedsiębiorstw. Pozwala na zwiększenie konkurencyjności przedsiębiorstw, przez wykorzystanie najnowszych osiągnięć badawczych w zakresie stosowania systemów informatycznych. Skompletowane zamówienia, w przypadku zamówień większych, są transportowane do miejsc przeznaczenia w kontenerach. Jedną z możliwości obniżenia kosztów transportu jest takie pakowanie przesyłek, aby maksymalnie zapełnić przestrzeń ładunkową kontenera. Zadaniem informatycznego systemu pakowania jest zaplanowanie

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie
** Wyższa Szkoła Biznesu, Dąbrowa Górnicza

rozmieszczenia przesyłek w kontenerze, które maksymalizuje upakowanie przesyłanego w nim towaru. Zwróćmy uwagę, że reguły gry rynkowej wymuszają na przedsiębiorstwach zajmujących się dystrybucją dóbr konsumpcyjnych realizacje zamówień detalicznych składających się z reguły z bardzo wielu małych produktów, co jest podstawową przyczyną dużego rozmiaru problemu pakowania. Z tego faktu wynika konieczność stosowania heurystycznych metod optymalizacji.

W rozdziale 2 artykułu dokonano przeglądu problemów pakowania, w 3 przedstawiono model matematyczny silnie NP-trudnego zagadnienia pakowania kontenera, punkt 4 zawiera definicję konturu wypukłego i zbioru punktów wstawienia oraz prezentuje ideę proponowanego rozwiązania dla zagadnień 2D i 3D, ostatni punkt jest podsumowaniem.

2. Przegląd problemów pakowania

Problematyka pracy wpisuje się w aktualny nurt zagadnień prezentowanych w literaturze światowej i dotyczy specyficznego elementu obsługi transportu polegającego na optymalizacji pakowania produktów, będącej ważnym składnikiem zarządzania łańcuchem dostaw i wspomaganiu zarządzania gospodarką magazynową.

Problemy optymalizacji pakowania zostały sklasyfikowane w publikacjach przez Dyckhoffa [6] oraz Dyckhoffa, Scheithauera i Terno [7]:

- *Strip packing* – w przypadku jeśli kontener posiada określoną szerokość oraz wysokość. Problem polega na upakowaniu w nim maksymalnej liczby prostokątów. Powyższy problem ma zastosowanie, między innymi w sytuacji, gdy ładunek musi być podzielony na rozdzielne sekcje przeznaczone dla różnych odbiorców. Zagadnienie to jest przedmiotem studiów Bischoffa i Ratcliffa w [2].
- *Knapsack loading* – zagadnienie pokrewne do zagadnienia plecakowego (*knapsack problem*). Różni się jedynie funkcją celu, która minimalizuje niewykorzystane miejsce w kontenerze. Algorytmy heurystyczne stosowane do rozwiązywania tego problemu można znaleźć m.in. publikacjach Gehringa, Menschera i Meyera [10] oraz Scheithauera [15].
- *Bin-packing* – problem pakowania do kontenerów o ustalonych wymiarach w celu minimalizacji liczby zastosowanych kontenerów. Zagadnienie to jest rozważane przez Scheithauera [16] oraz w pracy Faroe, Pisingera, Zachariasena [8]. W publikacji Martella, Pisingera oraz Vigo [12] zaproponowano dedykowany algorytm dokładny dla tego problemu.
- *Multi-container loading* – problem pokrewny do zagadnienia *bin-packing*, w którym rozważa się kontenery o różnych wymiarach. Celem jest taki ich dobór, aby zminimalizować koszt transportu. Powyższy problem optymalizacyjny jest rozważany przez Chena, Lee i Shena [4].

- *Knapsack container loading problem* – zagadnienie, w którym zakłada się, że zysk z załadowania pakunków jest proporcjonalny do ich wymiarów. Pakunki mogą być dowolnie obracane w ortogonalnych kierunkach. Celem jest upakowanie w kontenerze jak największej ich liczby, bez rozważania czy jest on rozłożony bezpiecznie lub umocowany [14].
- *Fixed-charge multiple knapsack problem* – jest rozszerzeniem zagadnienia *multiple knapsack* (MKP). Dla problemu tego zastosowano metodę podziału i ograniczeń [17], z dolnym i górnym ograniczeniem opartym o relaksację Lagrange'a wraz z zachłanną heurystyką i redukcją rozmiaru problemu. W wyniku tego, uzyskano algorytm dokładny pozwalający wydajnie rozwiązywać instancje opisujące problem pakowania 32 000 pakunków do 50 kontenerów.
- *Multiple knapsack problem* – zagadnienie polegające na lokowaniu pakunków do wielu opakowań, którego celem jest maksymalizacja zysku związanego z wybranymi pakunkami. Dla tego zagadnienia Pisinger [13] zaproponował algorytm podziału i ograniczeń, gdzie górne ograniczenie jest określane w oparciu o dedykowany algorytm programowania dynamicznego. W pracach Marello i Toth zaproponowano dla tego samego zagadnienia algorytm MTM [11].

Z praktycznego punktu widzenia największe znaczenie mają zagadnienia pakowania paczek prostopadłościennych. Wynika to z powszechności stosowania opakowań kartonowych oraz prostoty opisu, w stosunku do zagadnień pakowania przedmiotów o zróżnicowanym kształcie. Dla uproszczenia często zastępuje się bryły ich prostopadłościennym obrysem. Jedynie w przypadku jednorodnego asortymentu rozważa się praktycznie modele pakowania np.: przedmiotów kulistych, walcowych, czworościennych, itp. Powiązanym z zagadnieniem pakowania jest problem ładowania palet. Ponieważ palety są mniej stabilne, w stosunku do kontenerów, to w tym przypadku ważnym ograniczeniem jest wyważenie palety. Tu należy wspomnieć prace Bischoffa, Janetza oraz Ratcliffa [3], dotyczące ładowania na palety obiektów o różnych kształtach.

3. Model matematyczny trójwymiarowego zagadnienia pakowania

Rzeczywiste zagadnienie pakowania paczek, w znamienitej większości cytowanej literatury [1–17], sprowadzane jest do trójwymiarowego zagadnienia plecakowego (3DKP). Jest ono problemem *NP-trudnym – Pisinger* [14].

Dany jest tu zbiór n prostopadłościennych paczek $j = 1, ..., n$ o wymiarach: w_j, h_j, d_j oraz zysk związany z zapakowaniem paczki – c_j, do kontenera o wymiarach W, H, D. Sposób pakowania ma wykluczyć wzajemne „nachodzenie" na siebie paczek i wystawanie paczek poza obrys kontenera. Poszukujemy podzbioru zbioru paczek, który spełniając

ograniczenia przestrzeni ładunkowej, maksymalizuje sumaryczny zysk załadowanych paczek. W przypadku gdy zysk związany z paczką jest równy jej objętości maksymalizujemy wykorzystanie przestrzeni kontenera. Oprócz objętości wykorzystanego (wolnego) miejsca w kontenerze jako parametr oceny jakości rozwiązania zagadnienia pakowania stosuje się także:

- liczbę zapakowanych do kontenera paczek (dla każdej paczki $c_j = 1$),
- liczbę koniecznych kontenerów dla zapakowania wszystkich paczek,
- wyważenie kontenera – momenty sił liczone względem wybranej (wybranych) osi, położenie środka ciężkości.

Często dopuszcza się obracanie paczek – swobodne (zachowując równoległość do osi układu współrzędnych – ścian kontenera), które można zrealizować na 5 sposobów:

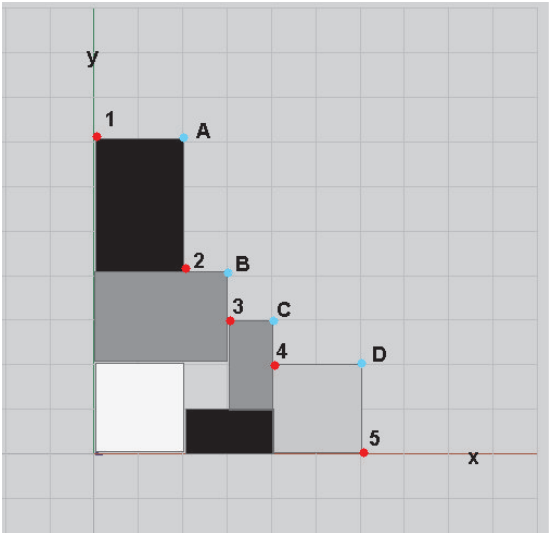
- rotacja 0 – bez obrotu,
- rotacja 1 – wokół osi x (odpowiada wymiarowi W),
- rotacja 2 – wokół osi y (odpowiada wymiarowi H),
- rotacja 3 – wokół osi z (odpowiada wymiarowi D),
- rotacja 4 – wokół osi x i y,
- rotacja 5 – wokół osi y i z,

lub ograniczone tylko do sposobu 2 (paczki oznaczone: „↑” – góra, nie wolno przewracać).

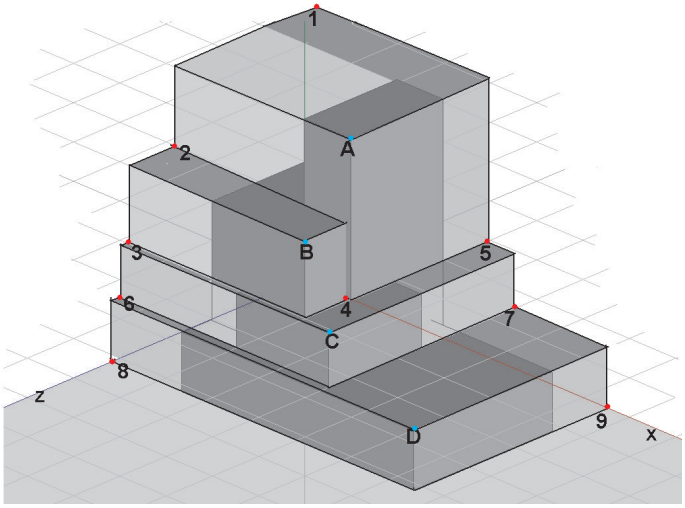
4. Kontur wypukły i zbiór punktów wstawienia

Pojęcie konturu (obwiedni) związane jest ze zrealizowanym rozwiązaniem częściowym i zdefiniowane przez zbiór charakterystycznych punktów, określający przestrzeń (powierzchnię – 2D) zajętą przez zapakowane paczki (określa stan związany z rozwiązaniem). Kontur (lista punktów – *CPL*) jest pomocniczą strukturą danych, tworzoną na podstawie aktualnego rozwiązania. Założeniem sposobu jego opisu jest możliwa prostota, stąd dążność do zmniejszenia liczby punktów definiujących kontur, a co za tym idzie, nakładu obliczeniowego związanego z jego obsługą (procedury obliczenia wartości funkcji celu, dodania – usunięcia punktów konturu, dodania – usunięcia punktów wstawienia). Konsekwencją przyjętego uproszczenia opisu, kosztem jakości tworzonych rozwiązań jest założenie wypukłości konturu. Kontur jest wypukły, jeśli odcinek łączący dwa dowolne jego punkty oraz równoległy do jednej z osi układu współrzędnych, zawiera się w konturze. Redukcja ilości danych związanych z opisem rozwiązania zagadnienia pakowania jest znaczna, w krańcowym przypadku rozwiązanie o praktycznie dowolnym rozmiarze możemy opisać za pomocą konturu zdefiniowanym np. jednym lub kilkoma punktami.

Kontur wypukły dla zagadnienia dwuwymiarowego (2D) ilustruje rysunek 1 – zbiór {A, B, C, D}, oraz dla zagadnienia trójwymiarowego (3D) rysunek 2 – zbiór {A, B, C, D}.



Rys. 1. Kontur wypukły – zbiór {A, B, C, D} oraz zbór punktów wstawienia {1, 2, ..., 5} dla dwuwymiarowego zagadnienia pakowania



Rys. 2. Kontur wypukły – zbiór {A, B, C, D} oraz zbór punktów wstawienia {1, 2, ..., 9} dla trójwymiarowego zagadnienia pakowania

Pomocniczą strukturą danych związaną z konturem jest także zbiór punktów wstawienia, definiujący miejsca potencjalnego dołożenia nowych paczek do aktualnego rozwiązania. Dla zbioru punktów wstawienia, zakładającego dosunięcie paczek do konturu,

wymagane jest dodatkowe sprawdzenie warunków ograniczeń zewnętrznych przestrzeni ładunkowej. Liczność zbioru punktów wstawienia dla konturu 2D (o liczności n) wynosi $n + 1$ (rys. 1 – zbiór {1, 2, ..., 5}), natomiast dla konturu 3D maksymalnie $2n + 1$ (rys. 2 – zbiór {1, 2, ..., 9}).

4.1. Procedura obliczenia objętości (pola) konturu 3D (2D)

Obliczenie pola konturu wypukłego 2D, zdefiniowanego przez listę punktów konturu, przebiega w oparciu o posortowanie listy według współrzędnych punktów. Procedura sumuje pola prostokątów, utworzonych dla kolejnych punktów, uwzględniając wcześniej wliczoną powierzchnię.

Objętość_konturu_2D (*CPL*, *n*)
Krok 1. $VC := 0$;
Posortuj *CPL* tak, aby: $x_i < x_{i+1}$ oraz $y_i > y_{i+1}$
Krok 2. $VC := x_1 * y_1$
Dla $i = 2, .., n$: $VC := VC + (x_i - x_{i-1}) * y_i$;
Krok 3. Zwróć *VC*;
gdzie:
CPL – lista punktów konturu,
n – długość listy *CPL*.

Procedura sortowania dla konturu 2D (pierwsza ćwiartka)

Procedura ustala kolejność listy punktów konturu, usuwając punkty nadmiarowe lub nie spełniające warunku wypukłości konturu. Wstawienie punktu I do posortowanej części *CPL* wymaga sprawdzenia poniższych warunków dla kolejnych jej punktów J:

jeżeli ($x_i > x_j$)
jeżeli ($y_i \geq y_j$) to usuń J z konturu;
inaczej (tzn. $y_i < y_j$) I wstaw po J;
jeżeli ($x_i = x_j$)
jeżeli ($y_i > y_j$) to usuń J z konturu;
inaczej (tzn. $y_i \leq y_j$) usuń I z konturu;
jeżeli ($x_i < x_j$)
jeżeli ($y_i > y_j$) to I wstaw przed J;
inaczej (tzn. $y_i \leq y_j$) usuń I z konturu;

W przypadku zdefiniowanego początkowego punktu wstawienia, punkt ten dzieli płaszczyznę na cztery ćwiartki. W k-tej ćwiartce ($k = 1, ..., 4$) punkty konturu rozpatruje się osobno, tworząc cztery listy $CPL[k]$ o różnej długości n_k.

Objętość_konturu_2D ($CPL[4]$, n_1, n_2, n_3, n_4, x_0, y_0)
Krok 1. $VC := 0$;
Posortuj $CPL[1]$ tak, aby: $x_i < x_{i+1}$ oraz $y_i > y_{i+1}$;
Posortuj $CPL[2]$ tak, aby: $x_i > x_{i+1}$ oraz $y_i > y_{i+1}$;
Posortuj $CPL[3]$ tak, aby: $x_i > x_{i+1}$ oraz $y_i < y_{i+1}$;
Posortuj $CPL[4]$ tak, aby: $x_i < x_{i+1}$ oraz $y_i < y_{i+1}$;
Krok 2. Dla kolejnych $CPL[k]$ (k = 1, ..., 4)
Dla i = 1, .., n_k: $VC := VC + |x_i - x_{i-1}| * |y_i - y_0|$;
Krok 3. Zwróć VC;

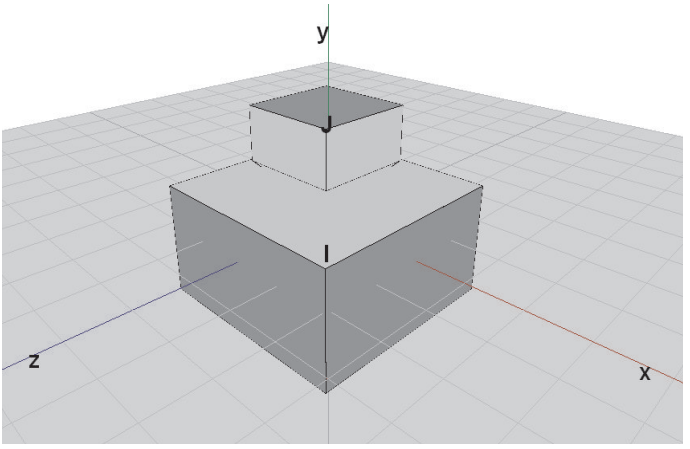
gdzie:

$CPL[k]$ – lista punktów konturu dla k-tej ćwiartki,
n_k – długość listy $CPL[k]$,
(x_0, y_0) – początkowy punkt wstawienia – punkt podziału na ćwiartki.

Procedura sortowania CPL dla konturu 3D – pierwszy oktant

Zależnie od przyjętej kolejności punktów *CPL* w trakcie obliczania objętości konturu możliwe jest zróżnicowanie nakładu obliczeniowego wynikające z podziału przestrzeni konturu na większą liczbę prostopadłościanów i konieczność wyznaczenia dodatkowych punktów (krawędzi, płaszczyzn) podziału. Za idealną należy uznać taką kolejność, dla której możliwy jest podział objętości konturu jedynie na prostopadłościany utworzone na bazie punktów *CPL*. Analiza kolejności rozpatrywanych punktów konturu wypukłego dla dwóch punktów prowadzi do 8 przypadków, które łącząc, można sformułować dla oktantu I następująco:

Przypadek 1 (rys. 3)



Rys. 3. Przypadek 1 wzajemnego położenia punktów konturu dla trójwymiarowego zagadnienia pakowania

Współrzędne punktów konturu: I(x_i, y_i, z_i) oraz J(x_j, y_j, z_j) spełniają warunki:

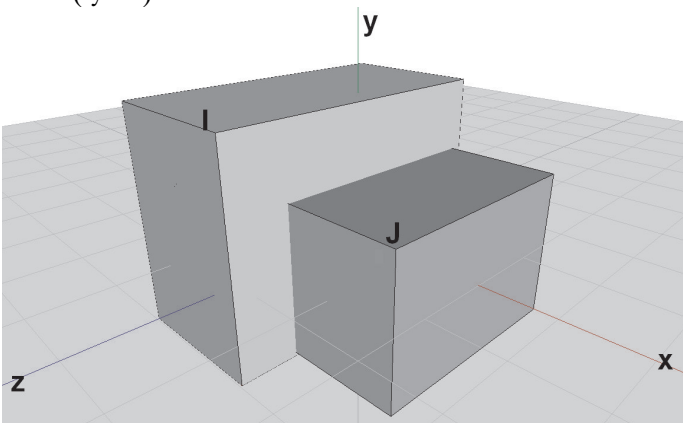
$$x_i > x_j, y_i < y_j, z_i > z_j$$

W obliczeniu objętości należy najpierw wziąć punkt I a potem J, co w sortowaniu listy *CPL* wymaga wstawienia I przed J. Objętość konturu wyraża się wzorem:

$$VC = (x_i - x_0)*(y_i - y_0)*(z_i - z_0) + (x_j - x_0)*(y_j - y_i)*(z_j - z_0)$$

Dla przypadku odwrotnego (tzn. $x_i < x_j$, $y_i > y_j$, $z_i < z_j$) I wstaw po J.

Przypadek 2 (rys. 4)



Rys. 4. Przypadek 2 wzajemnego położenia punktów konturu dla trójwymiarowego zagadnienia pakowania

Współrzędne punktów konturu: I(x_i, y_i, z_i) oraz J(x_j, y_j, z_j) spełniają warunki:

$$x_i < x_j, y_i > y_j, z_i > z_j$$

W sortowaniu listy *CPL* I należy wstawić przed J. Objętość konturu opisuje wzór:

$$VC = (x_i - x_0)*(y_i - y_0)*(z_i - z_0) + (x_j - x_i)*(y_j - y_0)*(z_j - z_0)$$

Dla przypadku odwrotnego (tzn. $x_i > x_j$, $y_i < y_j$, $z_i < z_j$) I wstaw po J.

Przypadek 3 (rys. 5)

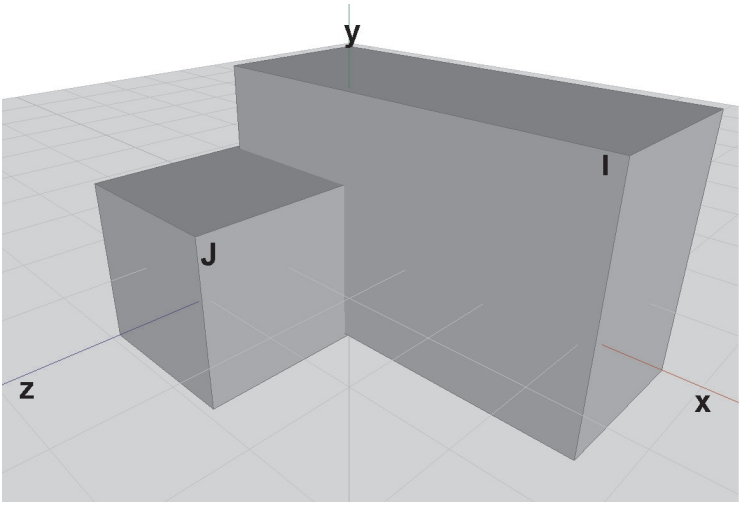
Współrzędne punktów konturu: I(x_i, y_i, z_i) oraz J(x_j, y_j, z_j) spełniają warunki:

$$x_i > x_j, y_i > y_j, z_i < z_j$$

W sortowaniu listy *CPL* I należy wstawić przed J. Objętość konturu opisuje wzór:

$$VC = (x_i - x_0)*(y_i - y_0)*(z_i - z_0) + (x_j - x_0)*(y_j - y_0)*(z_j - z_i)$$

Dla przypadku odwrotnego (tzn. $x_i < x_j$, $y_i < y_j$, $z_i > z_j$) I wstaw po J.



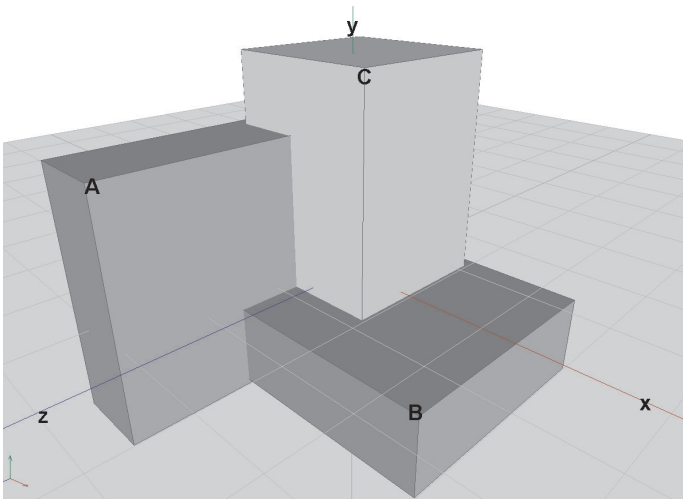
Rys. 5. Przypadek 3 wzajemnego położenia punktów konturu dla trójwymiarowego zagadnienia pakowania

Pozostałe przypadki obejmują sytuacje, gdy wszystkie współrzędne punktu I są większe od J (tzn. $x_i > x_j, y_i > y_j, z_i > z_j$) lub odwrotnie (tzn. $x_i < x_j, y_i < y_j, z_i < z_j$). Należy wtedy usunąć z konturu odpowiednio punkt J lub I. Przypadki, gdy niektóre ze współrzędnych są równe, uwzględniono w poniższej procedurze łączącej wymienione wcześniej sytuacje. Procedura polega na określeniu kolejności rozpatrywania punktów konturu, przez wstawienie kolejnego punktu I o współrzędnych (x_i, y_i, z_i) do posortowanej już części listy. Wstawienie punktu I przed lub po punkcie J określają warunki:

jeżeli ($x_i > x_j$)
  jeżeli ($y_i \geq y_j$)
    jeżeli ($z_i \geq z_j$) to usuń J z konturu;
    inaczej (tzn. $z_i < z_j$) I wstaw przed J;
  inaczej (tzn. $y_i < y_j$)
    jeżeli ($z_i \geq z_j$) to I wstaw przed J;
    inaczej (tzn. $z_i < z_j$) I wstaw po J;
jeżeli ($x_i = x_j$)
  jeżeli ($y_i \geq y_j$)
    jeżeli ($z_i \geq z_j$) usuń J z konturu;
    inaczej (tzn. $z_i < z_j$) I wstaw przed J;
  inaczej (tzn. $y_i < y_j$)
    jeżeli ($z_i > z_j$) to I wstaw po J;
    inaczej (tzn. $z_i \leq z_j$) usuń I z konturu;

jeżeli ($x_i < x_j$)
  jeżeli ($y_i > y_j$)
    jeżeli ($z_i \geq z_j$) to I wstaw przed J;
    inaczej (tzn. $z_i < z_j$) I wstaw po J;
  inaczej (tzn. $y_i \leq y_j$)
    jeżeli ($z_i > z_j$) to I wstaw po J;
    inaczej (tzn. $z_i \leq z_j$) usuń I z konturu

Zaproponowana metoda uszeregowania punktów konturu miała za zadanie obliczenie jego objętości w oparciu o prostopadłościany utworzone jedynie na podstawie punktów *CPL*, bez ich podziału na kilka prostopadłościanów (złożoność liniowa). Reguły opracowane dla dwóch punktów, choć okazały się skuteczne w większości zadań testowych o większej liczbie punktów, zawodzą w przypadku ogólnym.

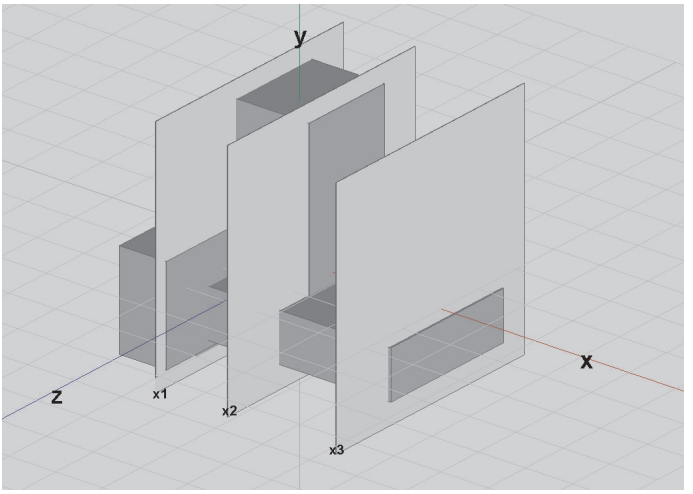


Rys. 6. Przykład „złośliwy” dla sortowania punktów konturu 3D

Przykładowe zadanie „złośliwe” jest zilustrowane na rysunku 6, gdzie kontur jest zdefiniowany przez 3 punkty o współrzędnych odpowiednio: A(1, 3, 5), B(4, 2, 3) i C(2, 4, 2). Dla tego przykładu punkt A winien być rozpatrywany przed B, B przed C oraz C przed A, czyli obliczenie objętości konturu jedynie przy podziale na 3 prostopadłościany nie jest możliwe. Konieczny jest podział na ich większą liczbę, a to wymaga wyznaczenia dodatkowych punktów pomocniczych na podstawie *CPL*.

Zastosowana ostatecznie metoda obliczenia objętości konturu polega na podziale przestrzeni konturu na warstwy wyznaczone przez punkty *CPL*, na podstawie jednej ze współrzędnych (w podanej poniżej procedurze wybrano współrzędną x). Punkty konturu są szeregowane według wybranej współrzędnej (lista *PLX*) oraz drugiej współrzędnej (np. y – lista

PLY) ustalającej kolejność punktów w dalszym podziale warstwy na prostopadłościany. Wybór współrzędnych jest dowolny, lecz determinuje on podział przestrzeni konturu płaszczyznami – w naszym przypadku równoległymi do *yz* (podział na warstwy – rys. 7) oraz *xy* (podział warstwy).



Rys. 7. Podział przestrzeni konturu na warstwy płaszczyznami równoległymi do *yz*

W obliczeniu objętości warstwy konturu uwzględniane są punkty *CPL* należące do warstwy, które są następnie usuwane z list *PLX* i *PLY*, oraz punkty spoza warstwy tworzące krawędzie konturu przechodzące przez warstwę (część punktów *PLY*).

Całkowita objętość konturu wypukłego jest sumą objętości prostopadłościanów wszystkich warstw. Warunkiem zakończenia obliczeń jest usunięcie wszystkich punktów z *PLX* oraz *PLY*.

Objętość_konturu_3D (*CPL*, n, x_0, y_0, z_0) – pierwszy oktant

Krok 1. $VC := 0$;

Posortuj *CPL*, według współrzędnej:

x – lista *PLX* – tak aby: $x_i \le x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i > y_{i+1}$

y – lista *PLY* – tak aby: $y_i \ge y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i < z_{i+1}$

Krok 2. Dla kolejnych punktów *PLX* ($i = 1, ..., n$)

$j^* := 0$;

$X := x_i - x_{i-1}$; (dla pierwszego $x_{i-1} = x_0$ – zadany punkt podziału lub $x_{i-1} = 0$)

Dla kolejnych punktów *PLY* (j)

Jeżeli ($x_j = x_i$)

– tzn. punkt konturu jest w aktualnie rozpatrywanej warstwie

to $Y := y_j - y_0$; (gdzie: y_0 – zadany punkt podziału lub $y_0 = 0$)

$Z := z_j - z_{j^*}$;

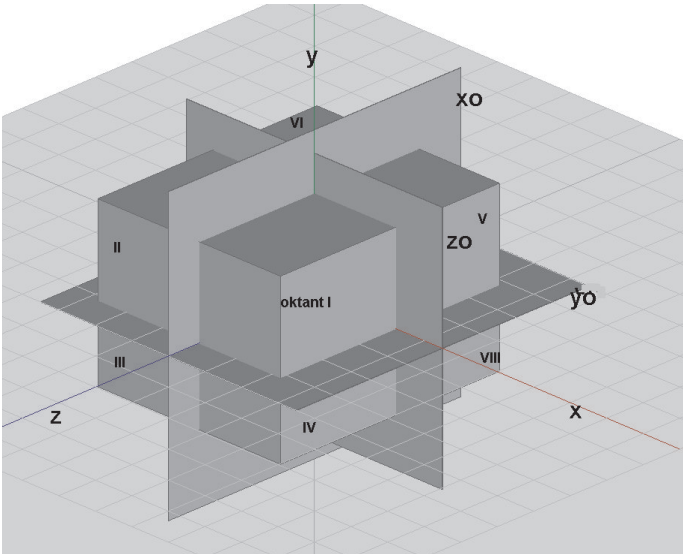
$VC := VC + X*Y*Z$;
$j^* := j$;
Usuń j z *PLX* oraz *PLY*;
inaczej (tzn. jeżeli $x_j > x_i$)
jeżeli ($z_j > z_{j^*}$) (tzn. gdy $z_j \leq z_{j^*}$ to j pomijamy)
to $Y := y_j - y_0$;
$Z := z_j - z_{j^*}$;
$VC := VC + X*Y*Z$;
$j^* := j$;

Krok 3. Zwróć *VC*;

gdzie:

CPL – lista punktów konturu,
n – długość listy *CPL*,
(x_0, y_0, z_0) – początkowy punkt wstawienia – punkt podziału na oktanty,
PLX – posortowana lista *CPL* według współrzędnej x,
PLY – posortowana lista *CPL* według współrzędnej y,
i, j – indeksy punktów konturu na liście *PLX* i *PLY*,
j^* – indeks poprzednio wykorzystanego punktu konturu.

Podział przestrzeni na oktanty (rys. 8) jest realizowany przez 3 płaszczyzny, wzajemnie prostopadłe, przecinające się w zadanym punkcie (x_0, y_0, z_0).



Rys. 8. Podział przestrzeni ładunkowej kontenera na oktanty

Z uwagi na brak dominującego standardu oznaczeń, oktanty ponumerowano następująco:

- oktant I: $x > x_0 , y > y_0 , z > z_0$
- oktant II: $x < x_0 , y > y_0 , z > z_0$
- oktant III: $x < x_0 , y < y_0 , z > z_0$
- oktant IV: $x > x_0 , y < y_0 , z > z_0$
- oktant V: $x > x_0 , y > y_0 , z < z_0$
- oktant VI: $x < x_0 , y > y_0 , z < z_0$
- oktant VII: $x < x_0 , y < y_0 , z < z_0$
- oktant VIII: $x > x_0 , y < y_0 , z < z_0$

W każdym oktancie punkty konturu rozpatruje się osobno, tworząc listy *CPL*[1], ..., *CPL*[8] o różnej długości n_1, ..., n_8.

Objętość_konturu_3D (*CPL*[8], *n*[8], x_0, y_0, z_0) – przypadek ogólny

Krok 1. *VC* := 0;

Posortuj *CPL*[1], według współrzędnej:
x – lista *PLX*[1] – tak aby: $x_i \le x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i > y_{i+1}$
y – lista *PLY*[1] – tak aby: $y_i \ge y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i < z_{i+1}$

Posortuj *CPL*[2], według współrzędnej:
x – lista *PLX*[2] – tak aby: $x_i \ge x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i > y_{i+1}$
y – lista *PLY*[2] – tak aby: $y_i \ge y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i < z_{i+1}$

Posortuj *CPL*[3], według współrzędnej:
x – lista *PLX*[3] – tak aby: $x_i \ge x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i < y_{i+1}$
y – lista *PLY*[3] – tak aby: $y_i \le y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i < z_{i+1}$

Posortuj *CPL*[4], według współrzędnej:
x – lista *PLX*[4] – tak aby: $x_i \le x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i < y_{i+1}$
y – lista *PLY*[4] – tak aby: $y_i \le y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i < z_{i+1}$

Posortuj *CPL*[5], według współrzędnej:
x – lista *PLX*[5] – tak aby: $x_i \le x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i > y_{i+1}$
y – lista *PLY*[5] – tak aby: $y_i \ge y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i > z_{i+1}$

Posortuj *CPL*[6], według współrzędnej:
x – lista *PLX*[6] – tak aby: $x_i \ge x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i > y_{i+1}$
y – lista *PLY*[6] – tak aby: $y_i \ge y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i > z_{i+1}$

Posortuj *CPL*[7], według współrzędnej:
x – lista *PLX*[7] – tak aby: $x_i \ge x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i < y_{i+1}$
y – lista *PLY*[7] – tak aby: $y_i \le y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i > z_{i+1}$

Posortuj *CPL*[8], według współrzędnej:
x – lista *PLX*[8] – tak aby: $x_i \le x_{i+1}$ oraz jeśli $x_i = x_{i+1}$ to $y_i < y_{i+1}$
y – lista *PLY*[8] – tak aby: $y_i \le y_{i+1}$ oraz jeśli $y_i = y_{i+1}$ to $z_i > z_{i+1}$

Krok 2. Dla kolejnych oktantów (k = 1, ..., 8)
Dla kolejnych punktów $PLX[k]$ (i = 1, ..., n_k)
$j^*:= 0$;
$X:= |x_i - x_{i-1}|$; (dla pierwszego $x_{i-1} = x_0$ – zadany punkt podziału lub $x_{i-1} = 0$)
Dla kolejnych punktów $PLY[k]$ (j)
Jeżeli ($x_j = x_i$)
– tzn. punkt konturu jest w aktualnie rozpatrywanej warstwie
to $Y:= |y_j - y_0|$; (gdzie: y_0 – zadany punkt podziału lub $y_0 = 0$)
$Z:= |z_j - z_{j^*}|$;
$VC: = VC+X*Y*Z$;
$j^*:= j$;
Usuń j z $PLX[k]$ oraz $PLY[k]$;
inaczej
Jeżeli ($z_j > z_{j^*}$ dla k=1,..,4) lub ($z_j < z_{j^*}$ dla k=5,..,8)
to $Y:= |y_j - y_0|$;
$Z:= |z_j - z_{j^*}|$;
$VC := VC+X*Y*Z$;
$j^*:= j$;
Krok 3. Zwróć VC;

gdzie:

$CPL[k]$	–	lista punktów konturu w k-tym oktancie,
n_k	–	długość listy $CPL[k]$,
(x_0, y_0, z_0)	–	początkowy punkt wstawienia – punkt podziału na oktanty,
$PLX[k]$	–	posortowana lista $CPL[k]$ według współrzędnej x,
$PLY[k]$	–	posortowana lista $CPL[k]$ według współrzędnej y,
i, j	–	indeksy punktów konturu na liście $PLX[k]$ i $PLY[k]$,
j^*	–	indeks poprzednio wykorzystanego punktu konturu.

W implementacji procedury obliczenia objętości konturu można stosować każdorazowo procedurę sortowania wszystkich punktów konturu, albo w przypadku dodania jednego punktu procedurę wstawienia punktu do posortowanej listy (CPL, PLX, PLY). Zmiana dotyczy tylko jednego oktantu, co umożliwia uproszczenie obliczenia objętości. Dodanie nowego punktu konturu może spowodować także usunięcie jednego lub kilku zbędnych punktów definiujących kontur wypukły. Współrzędne nowego punktu konturu (x_N, y_N, z_N) są wyznaczane na podstawie współrzędnych punktu wstawienia (x_W, y_W, z_W), wymiarów paczki: szerokość, wysokość, głębokość – z uwzględnieniem możliwego obrotu paczki – x_P, y_P, z_P oraz ograniczeń przestrzeni ładunkowej – x_{MIN}, y_{MIN}, z_{MIN} , x_{MAX}, y_{MAX}, z_{MAX}.

Dla kolejnych oktantów:

– oktant I: $x_N = x_W + x_P \le x_{MAX}$, $y_N = y_W + y_P \le y_{MAX}$, $z_N = z_W + z_P \le z_{MAX}$,
– oktant II: $x_N = x_W - x_P \ge x_{MIN}$, $y_N = y_W + y_P \le y_{MAX}$, $z_N = z_W + z_P \le z_{MAX}$,

– oktant III: $x_N = x_W - x_P \geq x_{MIN}$, $y_N = y_W - y_P \geq y_{MIN}$, $z_N = z_W + z_P \leq z_{MAX}$,
– oktant IV: $x_N = x_W + x_P \leq x_{MAX}$, $y_N = y_W - y_P \geq y_{MIN}$, $z_N = z_W + z_P \leq z_{MAX}$,
– oktant V: $x_N = x_W + x_P \leq x_{MAX}$, $y_N = y_W + y_P \leq y_{MAX}$, $z_N = z_W - z_P \geq z_{MIN}$,
– oktant VI: $x_N = x_W - x_P \geq x_{MIN}$, $y_N = y_W + y_P \leq y_{MAX}$, $z_N = z_W - z_P \geq z_{MIN}$,
– oktant VII: $x_N = x_W - x_P \geq x_{MIN}$, $y_N = y_W - y_P \geq y_{MIN}$, $z_N = z_W - z_P \geq z_{MIN}$,
– oktant VII: $x_N = x_W + x_P \leq x_{MAX}$, $y_N = y_W - y_P \geq y_{MIN}$, $z_N = z_W - z_P \geq z_{MIN}$.

Dodanie nowego punktu do konturu *CPL* eliminuje ze zbioru punktów wstawienia *IPL* (o liczności *m*) wykorzystany punkt oraz inne punkty wstawienia, które znalazły się w przestrzeni konturu wypukłego. Mogą natomiast powstać nowe punkty wstawienia. Ich liczba jest uzależniona od liczby usuniętych punktów *CPL* oraz *IPL*.

Aktualizacja_konturu (*CPL*, *n*, x_N, y_N, z_N, *IPL*, *m*) – aktualizacja listy punktów konturu oraz zbioru punktów wstawienia dla pierwszego oktantu

Krok 1. Dla każdego punktu konturu *CPL* (i = 1, .., n)
Jeżeli ($x_N \geq x_i \wedge y_N \geq y_i \wedge z_N \geq z_i$) to usuń punkt ($x_i$, y_i, z_i) z *CPL*;
Krok 2. Dodaj punkt (x_N, y_N, z_N) do *CPL* (aktualizacja n);
Krok 3. Dla każdego punktu zbioru punktów wstawienia *IPL* (j = 1, .., m)
Jeżeli ($x_N \geq x_j \wedge y_N \geq y_j \wedge z_N \geq z_j$) to usuń punkt ($x_j$, y_j, z_j) z *IPL*;
Krok 4. Uzupełnij *IPL* o nowopowstałe punkty wstawienia (aktualizacja m).

4.2. Idea algorytmu

Schemat rozwiązywania trójwymiarowego zagadnienia pakowania paczek, z wykorzystaniem konturu wypukłego i zbioru punktów wstawienia:

Krok 1. Wczytanie danych:

– wymiary paczek (kolejność paczek dowolna (losowa) lub określona według reguły zachłannej, możliwość stosowania rotacji – podane wymiary po dokonaniu wcześniejszej rotacji lub decyzja o rotacji dokonywana w trakcie pakowania)
– wymiary przestrzeni ładunkowej (kontenera) – określenie ograniczeń x_{MIN}, y_{MIN}, z_{MIN}, x_{MAX}, y_{MAX}, z_{MAX}.
– początkowy punkt wstawienia (x_0, y_0, z_0), dzielący przestrzeń ładunkową na oktanty ((0, 0, 0) – typowe zagadnienie pakowania od lewego, dolnego, tylnego rogu kontenera; (($x_{MAX} - x_{MIN}$)/2, 0, 0) – możliwość wyważenia kontenera wg jednej z osi; (($x_{MAX} - x_{MIN}$)/2, 0, ($z_{MAX} - z_{MIN}$)/2) – możliwość wyważenia kontenera wg dwóch osi)

Krok 2. Inicjowanie struktur (puste):

– rozwiązanie (wg przyjętego sposobu zapisu rozwiązania: np. permutacja numerów paczek – przy zadanym deterministycznym sposobie ich pakowania, permutacja paczek z rotacją lub numer paczki, rotacja, współrzędne punktu wstawienia paczki (x_W, y_W, z_W) – lewy, dolny, tylny róg)

– lista punktów konturu $CPL[k]$ – osobna dla każdego z oktantów k:
  $CPL[k] := (x_0, y_0, z_0)$;
– lista punktów wstawienia $IPL[k]$ – osobna dla każdego z oktantów k:
  $IPL[k] := (x_0, y_0, z_0)$;

Krok 3. Dla wybranej paczki:
(kolejność zadana lub paczka wybrana wg określonego kryterium)

– wybierz punkt wstawienia (x_W, y_W, z_W) z *IPL* (pierwszy, przegląd częściowy wg określonego kryterium, przegląd zupełny – spełnienie ograniczeń, możliwy obrót paczki)
– uzupełnij rozwiązanie (dopisz wstawioną paczkę)
– zaktualizuj listę punktów konturu $CPL[k]$ w wybranym oktancie (oblicz i dodaj nowy punkt konturu, usuń zbędne punkty *CPL*)
– zaktualizuj listę punktów wstawienia $IPL[k]$ w wybranym oktancie (usuń wykorzystany punkt wstawienia, dodaj nowopowstałe punkty, ewentualnie usuń punkty z wnętrza konturu wypukłego)
– oblicz funkcję celu (z wykorzystaniem objętości konturu – tylko w wybranym oktancie: np. objętość konturu – objętość wstawionych paczek)

Krok 4. Powtarzaj krok 3 do spełnienia warunku zakończenia algorytmu
(np. zapakowanie wszystkich paczek, brak możliwości wstawienia paczki, uzyskanie zadanej jakości rozwiązania)

5. Podsumowanie

– Zastosowanie konturu wypukłego znacznie upraszcza opis przestrzeni zajętej w stosunku do pełnej postaci rozwiązania, w której podane są lub obliczane współrzędne (oraz wymiary) wszystkich paczek. Następuje tu znaczna redukcja rozmiaru danych, jednak kosztem jakości rozwiązań związanych z założoną wypukłością konturu.
– Podział przestrzeni ładunkowej na oktanty (3D) lub ćwiartki (2D) pozwala realizować obliczenia w każdym oktancie (ćwiartce) niezależnie, dla mniejszego rozmiaru zagadnienia (dekompozycja).
– Liniowa złożoność obliczeniowa wyznaczenia pola konturu dla zagadnienia 2D nie jest możliwa w ogólnym przypadku objętości dla zagadnienia 3D (patrz przykład).
– Zaproponowane podejście jest dedykowane dla problemów o dużym rozmiarze, gdyż redukcja i dekompozycja znacznie ułatwia ich rozwiązanie.

Literatura

[1] Bansal N., Sviridenko M., *New approximability and inapproximability results for 2-dimensional Bin Packing*. Proc. of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, 2004.

[2] Bischoff E., Ratcliff M.S.W., *Loading multiple pallets*. The Journal of the Operational Research Society, vol. 46, 11, 1995, 1322–1336.
[3] Bischoff E., Janetz F., Ratcliff M.S.W., *Loading pallets with non-identical items*. European Journal of Operational Research 84, 1995, 681–692.
[4] Chen C.S., Lee S.M., Shen Q.S., *An analytical model for the container loading problem*. European Journal of Operational Research, 80, 1, 1995, 68–76.
[5] Dembo R.S., Hammer P.L., *A reduction algorithm for knapsack problems*. Methods of Operations Research, 36, 1980, 49–60.
[6] Dyckhoff H., *A typology of cutting and packing problems.* European Journal of Operational Research, 44, 1990, 145–159.
[7] Dyckhoff H., Scheithauer G., Terno J., *Cutting and Packing* (*C&P*). In: M. Dell'Amico, F. Maffioli, and S. Martello, editors, Annotated Bibliographies in Combinatorial Optimization. John Wiley & Sons, Chichester, 1997.
[8] Faroe O., Pisinger D., Zachariasen M., *Guided Local Search for the Three-Dimensional Bin Packing Problem*. INFORMS Journal on Computing, 1999.
[9] Gehring H., Bortfeld A., *A genetic algorithm for solving the container loading problem.* International Transactions of Operational Research, 4, 1997.
[10] Gehring M., Menscher K. Meyer M., *A computer-based heuristics for packing poole shipment containers*. EJOR, 44, 1990, 277–288.
[11] Marello S., Toth P., *Knapsack Problem: Algorithms and computer implementation*. Wiley, Chichester, 1990.
[12] Martello S., Pisinger S., Vigo D., *The Three-Dimensional Bin Packing Problem.* Operations Research, 48, 2, 2000, 256–267.
[13] Pisinger D., *An exact algorithm for large multiple knapsack problems*. European Journal of Operational Research, 114, 1999, 528–541.
[14] Pisinger D., *Heuristics for the container loading problem*. European Journal of Operational Research, 141, 2, 2002, 382–392.
[15] Scheithauer G., *Heuristics for the container loading problem*. European Journal of Operational Research, Springer-Verlag, 1991.
[16] Scheithauer G., *A three dimensional bin packing algorithm*. Journal of Information Processing and Cybernetics, 27, 1991, 263–271.
[17] Yamada T., Takeoka T., *An exact algorithm for the fixed-charge multiple knapsack problem*. European Journal of Operational Research, 192, 2, 2009, 700–705.