Wojciech Horzelski[1], Dariusz Doliwa[1], Mariusz Frydrych[2]
[1]Uniwersytet Łódzki, Katedra Informatyki Stosowanej,
Wyższa Szkoła Informatyki w Łodzi
email: horzel@math.uni.lodz.pl, doliwa@math.uni.lodz.pl
[2]Uniwersytet Łódzki, Katedra Metodyki Nauczania Matematyki,
Wyższa Szkoła Informatyki w Łodzi
email: frydrych@math.uni.lodz.pl

ANALIZA WSPÓŁCZYNNIKA STRATY ALGORYTMÓW PAKOWANIA FIRST FIT ORAZ BEST FIT PRZY ŁADUNKACH O ROZKŁADZIE GAUSSA

Streszczenie – Zadanie pakowania w klasycznym ujęciu polega na rozmieszczeniu listy ładunków L={a_1,a_2,…,a_n} o rozmiarach nieprzekraczających 1 w minimalnej ilości pojemników o rozmiarze jednostkowym, wymaga się przy tym, aby żaden z pojemników nie był przeładowany. Jest to zadanie z klasy problemów NP-trudnych. Do klasycznych algorytmów dla takiego problemu on-line należą metody First Fit oraz Best Fit. W pracy przedstawiono wyniki badań zachowania się współczynnika straty dla tych algorytmów przy założeniu, iż ładunki mają rozkład normalny na przedziale (0,1].

1 Sformułowanie problemu

Zadanie pakowania polega na rozmieszczeniu listy ładunków L = (a_1, …, a_n) , gdzie $a_i \in (0,1]$, w minimalnej ilości pojemników o rozmiarze 1. Wymaga się przy tym, aby suma zadań zapakowanych do każdego pojemnika nie przekraczała 1. Tak postawiony problem posiada różnorakie praktyczne zastosowania poczynając od logistycznych (np. optymalnego zapełnianie kontenerów) aż po zastosowania w sieciach komputerowych (równomierne obciążanie łączy w sieciach rozległych) [1].

DEFINICJA 1. Niech *J* będzie dowolnym skończonym podzbiorem zbioru liczb naturalnych i niech | *J* | oznacza moc tego zbioru.

Oznaczmy $\mathcal{L}_n = \bigcup_{J \subset N, |J|=n} I^J$ oraz $\mathcal{L} = \bigcup_{n=1}^{\infty} \mathcal{L}_n$.

Element $L \in \mathcal{L}$ nazywamy listą, jeżeli $L \in \mathcal{L}_n$ to mówimy, że lista ma długość n i oznaczamy |L|= n. Jeżeli $L \in I^J$, to zbiór *J* nazywamy zbiorem indeksów listy L i oznaczamy D_L, gdzie $I=(0,1]$

Oznaczmy przez S_L sumę wszystkich elementów listy: $\sum_{i \in D_L} L(i)$

Niech J_k oznacza zbiór {1, 2, … , k}$\subset N$.

DEFINICJA 2. Niech B będzie dowolnym zbiorem skończonym k-elementowym rozbiciem zbioru B nazywamy funkcję: $P_k^B : J_k \to 2^B$ spełniającą następujące warunki:

1. $\forall_{i \in J_k} P_k^B(i) \neq \emptyset$
2. $\forall_{i,j \in J_k\, i \neq j} P_k^B(i) \cap P_k^B(j) = \emptyset$
3. $\bigcup_{i \in J_k} P_k^B(i) = B$.

Liczbę *k* nazywamy mocą rozbicia P_k^B.

Oznaczmy przez $\wp_K^B$ podzbiór zbioru $(2^B)^{Jk}$ składający się ze wszystkich k-elementowych rozbić zbioru B, zaś przez $\wp^B$ zbiór $\bigcup_{k=1}^{|B|} \wp_k^B$.

Niech dalej $\wp^{(n)}$ oznacza zbiór $\bigcup_{J \subset N, |J|=n} \wp^J$, zaś $\wp$ zbiór $\bigcup_{i=1}^{\infty} \wp^{(n)}$.

DEFINICJA 3. Algorytmem pakowania A nazywamy odwzorowanie $A : \mathcal{L} \to \wp$ spełniające następujące warunki:

1. $A(L) \in \wp^{D_L}$;
2. $\forall_{j \in J_{|A(L)|}} \sum_{i \in A(L)(j)} L(i) \leq 1$.

Wartość |A(L)| nazywamy rezultatem (wynikiem) działania algorytmu A dla listy L.
Oznaczamy przez A zbiór wszystkich algorytmów pakowania.
Ze względu na zastosowania praktyczne elementy listy L nazywać będziemy dalej zadaniami, natomiast zbiory otrzymane w wyniku rozbicia pojemnikami. Będziemy mówić też, że zadanie L(*i*) zostało zapakowane algorytmem A do *j*-ego pojemnika, jeżeli $i \in A(L)(j)$.

DEFINICJA 4. Niech A będzie algorytmem pakowania i niech $L \in \mathcal{L}$. Poziomem pojemnika A(L)(j) nazywamy sumę elementów listy L, których indeksy należą do tego pojemnika $\sum_{i \in A(L)(j)} L(i)$ i oznaczamy lev(A(L)(j)).

Oznaczmy przez Sub(L) zbiór wszystkich podlist listy $L \in L$. Zauważmy, że $D_{L_j} = J$.

DEFINICJA 5. Algorytm pakowania A nazywamy on-line, jeżeli dla dowolnej listy $L \in \mathcal{L}$ zachodzi:

$$\forall_{L_j \in Sub(L)} \forall_{j \in J \setminus A(L_j)} A(L_j)(j) \subset A(L)(j).$$

Zgodnie z powyższą definicją algorytm on-line pakuje elementy listy L zgodnie z kolejnością indeksów.

DEFINICJA 6. Niech $A \in A$. Stratą algorytmu pakowania A nazywamy różnicę

$$|A(L)| - \sum_{i=1}^{|A(L)|} lev(A(L))(i))$$

i oznaczamy ją $WS_A(L)$.

DEFINICJA 7. Niech $A \in A$. Współczynnikiem straty dla algorytmu pakowania nazywamy stosunek straty algorytmu doi ości wykorzystanych pojemników

$$\frac{WS_A(L)}{|A(L)|}$$

i oznaczamy ją $R_{WS_A(A)}$.

2 Algorytmy First Fit Oraz Best Fit

Metoda First Fit (FF). W metodzie tej nie zamykamy pojemników, aż do zakończenia pakowania. Nowe zadanie umieszczane jest w pierwszym z pojemników, do którego się mieści [2]. Algorytm ten można zdefiniować poprzez indukcję względem długości pakowanej listy zadań L:

1. jeżeli |L| = 1, to $|FF(L)| = D_L$ i $FF(L)(1) = L$;
2. jeżeli zdefiniowane jest działanie algorytmu dla list o długości n – 1 i |L| = n, wtedy:

niech $L_{n-1} \in Sub(L) \wedge |L_{n-1}| = n-1$.

Jeżeli

$$\min_{k \in J_{|FF(L_{n-1})|}} (lev(FF(L_{n-1})(k))) + L(D_L \setminus D_{L_{n-1}}) > 1,$$

to $|FF(L)| = |FF(L_{n-1})| + 1$

i

$$\forall_{j \in |FF(L_{n-1})|} FF(L)(j) = FF(L_{n-1})(j)$$

oraz

$$FF(L)(|FF(L_{n-1})|+1) = D_L \setminus D_{L_{n-1}};$$

jeżeli

$$\min_{k \in J_{|FF(L_{n-1})|}} (lev(FF(L_{n-1})(k))) + L(D_L \setminus D_{L_{n-1}}) \leq 1$$

to

$$|FF(L)| = |FF(L_{n-1})|$$

i jeśli

$$k = \min\{i \in J_{|FF(L_{n-1})|} : lev(FF(L_{n-1})(i)) + L(D_L \setminus D_{L_{n-1}}) \leq 1\},$$

to

$$FF(L)(k) = FF(L_{n-1})(k) \cup D_L \setminus D_{L_{n-1}}$$

oraz

$$\forall_{i \in J_{|FF(L_{n-1})|} \wedge i \neq k} FF(L)(i) = FF(L_{n-1})(i).$$

Inaczej mówiąc, pakując zadanie szukamy wśród otwartych już pojemników takich, które mogą pomieścić to zadanie i wybieramy ten o najniższym indeksie. Jeżeli nie ma pojemników mogących pomieścić zadanie to otwierany jest nowy pojemnik, do którego pakowane jest rozpatrywane zadanie. Wszystkie pojemniki pozostają otwarte aż do wyczerpania się listy zadań.

Algorytm Best Fit (BF)

Zasada działania zbliżona jest do działania algorytmu First Fit. Różnica polega na tym, że Best Fit stara się wybrać ten z pojemników mieszczących element, który po dodaniu go będzie miał poziom zapakowania najbliższy jedności [2]. Dokładniej algorytm ten zdefiniujemy indukcyjnie ze względu na długość pakowanej listy zadań L:

1. jeżeli |L| = 1, to $|BF(L)| = D_L$ i BF(L)(1) = L;
2. jeżeli zdefiniowane jest działanie algorytmu dla list o długości $n-1$ i $|L| = n$, to:

Niech

$$L_{n-1} \in Sub(L) \wedge |L_{n-1}| = n-1 .$$

Jeżeli

$$\min_{k \in J_{|BF(L_{n-1})|}} (lev(BF(L_{n-1})(k))) + L(D_L \setminus D_{L_{n-1}}) > 1,$$

to

$$|BF(L)| = |BF(L_{n-1})| + 1$$

i

$$\forall_{j \in |BF(L_{n-1})|} BF(L)(j) = BF(L_{n-1})(j)$$

oraz

$$BF(L)(|BF(L_{n-1})| + 1) = D_L \setminus D_{L_{n-1}};$$

jeżeli

$$\min_{k \in J_{|BF(L_{n-1})|}} (lev(BF(L_{n-1})(k))) + L(D_L \setminus D_{L_{n-1}}) \leq 1,$$

to

$$|BF(L)| = |BF(L_{n-1})|$$

i jeśli $k \in J_{|BF(L_{n-1})|}$ jest takie, że

$$lev(BF(L_{n-1})(k)) = \max_{i \in J_{|BF(L_{n-1})|} \wedge lev(BF(L_{n-1})(i)) + L(D_L \setminus D_{L_{n-1}}) \leq 1)} lev(BF(L_{n-1})(i)),$$

to

$$BF(L)(k) = BF(L_{n-1})(k) \cup D_L \setminus D_{L_{n-1}}$$

oraz

$$\forall_{i \in J_{|BF(L_{n-1})|} \wedge i \neq k} BF(L)(i) = BF(L_{n-1})(i).$$

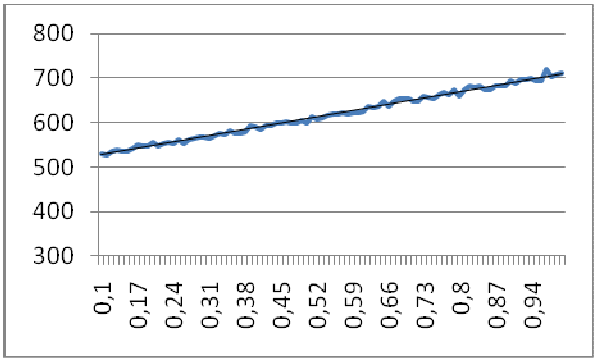
3 Analiza straty algorytmu First Fit przy zadaniach o rozkładzie Gaussa

Zachowanie się algorytmu First Fit zostało zbadane jedynie dla zadań o rozkładach dyskretnych[3] oraz dla zadań o rozkładzie jednostajnym na przedziale jednostkowym[2]. Niewiele wiadomo natomiast na temat zachowania się współczynnika straty dla innych ciągłych rozkładów.

Do badania wybrano rozkład normalny, jako że w praktycznych zastosowaniach zadania najczęściej spotykamy zadania o takim właśnie rozkładzie. Zastosowano tutaj generator liczb pseudolosowych oparty o metodę biegunową - Monte-Carlo [4,5].

Badania zostały przeprowadzone dla list o ustalonej długości (prezentowane poniżej wyniki dotyczą list o długości 1000 elementów).

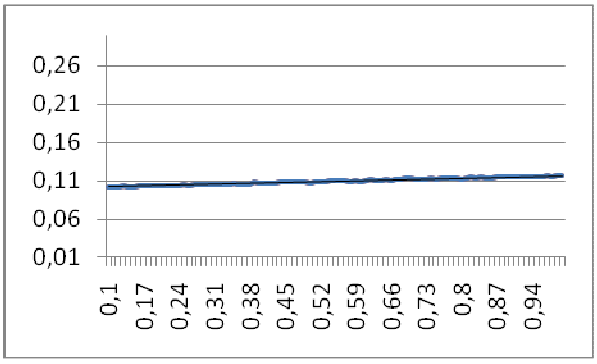
Wartości straty oraz współczynnika straty dla list o ustalonej długości są funkcją wartości oczekiwanej dla rozkładu elementu. Okazuje się, że jego zmiany w zależności od wartości oczekiwanej rozkładu mają charakter liniowy.



Rys. 1. Zależność straty algorytmu FF od wartości oczekiwanej rozkładu (lista o długości 10000 elementów, rozkład Gaussa)

Zależność tę można przybliżyć funkcją liniową wartości oczekiwanej rozkładu (dokładnie y = 2,01 E + 525,77, przy R-kwadrat = 0.994).

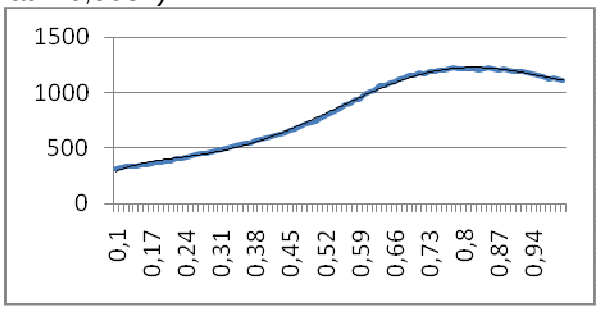
Analogicznie dla współczynnika straty mamy:



Rys. 2. Zależność współczynnika straty algorytmu FF od wartości oczekiwanej rozkładu (lista o długości 10000 elementów, rozkład Gaussa)

Jest to też zależność o charakterze liniowym, bliskim funkcji stałej (można ją aproksymować poprzez y = 0,0002 E + 0,1016, przy R-kwadrat = 0,98)

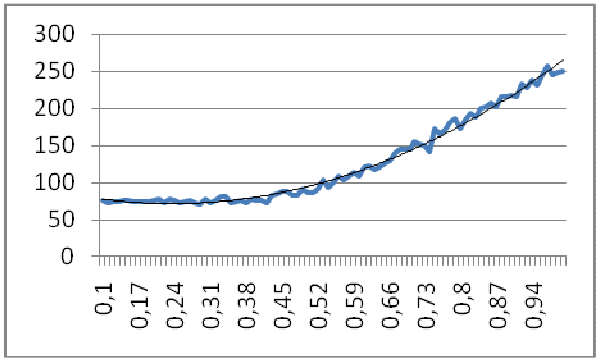
Dla porównania przy rozkładzie trójkątnym podobna zależność ma charakter wielomiany piątego stopnia:
$y = 3E\text{-}06E^5 - 0.0006E^4 + 0.0481E^3 - 1.346E^2 + 21.593E + 268.25$ (przy R-kwadrat = 0,9984)



Rys. 3. Zależność straty algorytmu FF od wartości oczekiwanej rozkładu (lista o długości 10000 elementów, rozkład trójkątny)

4 Analiza straty algorytmu Best Fit przy zadaniach o rozkładzie Gaussa

Podobnie jak w przypadku algorytmu First Fit zachowanie się algorytmu Best Fit dla list zadań o rozkładach innych niż jednostajny jest słabo zbadane. Pokazane zostało jedynie ograniczenie współczynnika straty dla rozkładów skośnych [6]. W praktycznych zadaniach spotyka się jednak najczęściej listy zadań rozkładzie normalnym. Współczynnik straty algorytmu Best Fit przy takim rozkładzie zadań jest kwadratowo zależny od zmian wartości oczekiwanej rozkładu (przy ustalonej długości listy L):



Rys. 4. Zależność straty algorytmu BF od wartości oczekiwanej rozkładu (lista o długości 10000 elementów, rozkład Gaussa)

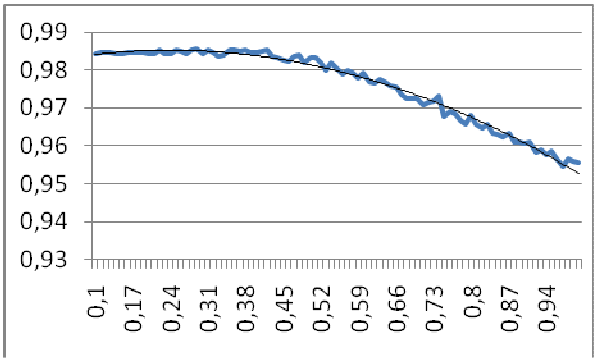
Dokładnie funkcję tę można aproksymować poprzez:

$$y = 0{,}0332E^2 - 0{,}9771E + 78{,}578$$

(przy R-kwadrat = 0.9904) .

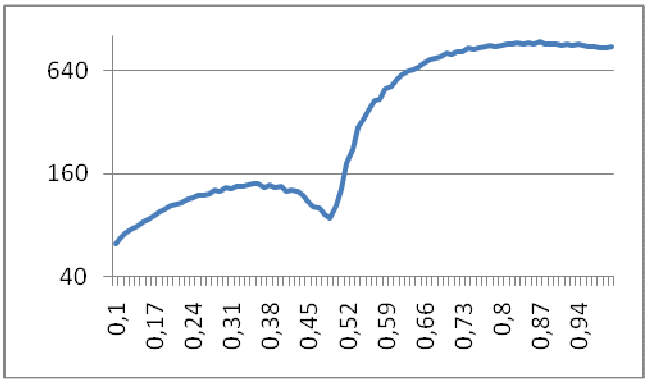
Czyli dla niewielkich wartości oczekiwanych (do ok. 0.3) algorytm Best Fit zachowuje się bardzo stabilnie utrzymując straty na niskim poziomie. Wartości współczynnika straty również mogą zostać przybliżone wielomianem drugiego stopnia:

$y = -6E\text{-}06x^2 + 0{,}0002x + 0{,}9839$ (przy R-kwadrat = 0,9872)



Rys. 5. Zależność współczynnika straty algorytmu BF od wartości oczekiwanej rozkładu (lista o długości 10000 elementów, rozkład Gaussa)

Podobnie jak w przypadku algorytmu First Fit warto porównać to zachowanie się z innym rozkładem o podobnym charakterze. Dla tego celu wybrano rozkład trójkątny ze zmienną wartością oczekiwaną:



Rys. 6. Zależność straty algorytmu BF od wartości oczekiwanej rozkładu (lista o długości 10000 elementów, rozkład trójkątny)

5 Wnioski

Przedstawione powyżej wyniki świadczą o stabilnym zachowaniu się współczynnika nadwyżki algorytmów First Fit oraz Best Fit dla list ładunków o rozkładzie normalnym na przedziale (0,1].

W metodzie First Fit współczynnik ten można opisać funkcją liniową wartości oczekiwanej wielkości ładunków, bliską funkcji stałej. Inaczej mówiąc metoda zachowuje się podobnie niezależnie od tego czy przeważać będą duże czy małe zadania.

Inaczej jest w przypadku algorytmu Best Fit. Dla list z przewagą małych zadań (dla list o rozkładach z wartością oczekiwaną ≤ 0,35) współczynnik straty ma niemalże wartość stałą. Dla list o większych zadaniach metoda pakuje zadania coraz bardzie efektywnie (osiągając współczynniki straty na poziomie 0,96-0,95). Metoda Best Fit podobnie jak dla rozkładu jednostajnego okazuje się bardziej efektywna, okupione jest to oczywiście zwiększonym czasem wykonania procesu pakowania [2].

Literatura

[1] Lee C., Lee D., A simple packing algorithm, Journal of the ACM 32, 1985, 562-575

[2] Hoffman E.G., Leuker G.S., *Probabilistic Analysis of Packing and Partitioning Algorithms,* John Wiley &Sons, New York 1991

[3] Chan, Wun-Tat; Lam, Tak-Wah; Wong, Prudence W. H. Dynamic bin packing of unit fractions items. Automata, languages and programming, 614--626

[4] Karg, Ch., Köbler J., Schuler R., *The complexity of generating test instances.* STACS 97 (Lübeck), 375--386, Lecture Notes in Comput. Sci., 1200, Springer, Berlin, 1997.

[5] Knuth D. E., The Art of Computer Programming - Seminumerical Algorithms, Addison-Wesley 1997, 1998

[6] Kenyon C., Mitzenmacher, Michael Linear waste of best fit bin packing on skewed distributions. Probabilistic methods in combinatorial optimization. Random Structures Algorithms 20 (2002), no. 3, 441--464. Analysis of Algorithms for Bin Packing Problems, *Journal of Algorithms* 12 (1991), 189-203

ANALYSIS OF THE LOSS RATE OF FIRST FIT AND BEST FIT PACKING ALGORITHMS BY CHARGES WITH GAUSS DISTRIBUTION

Summary - The task of packing in the classic take consists in laying the list of L= cargoes out {and 1, and 2, …, an} about not exceeding sizes 1 in the minimal number of containers about the individual size, they demand in addition that none of containers is overloaded. There is this assignment on the class of problems NP-trudnych. For such a problem on-line First Fit methods and Best Fit belong to classic algorithms.