Wojciech Horzelski
Uniwersytet Łódzki, Katedra Informatyki Stosowanej,
Wyższa Szkoła Informatyki w Łodzi

ANALIZA PROBABILISTYCZNA WYBRANYCH SEKWENCYJNYCH ALGORYTMÓW PAKOWANIA

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. Wśród metod rozwiązania takiego zadania jest też klasa algorytmów sekwencyjnych. Od algorytmu sekwencyjnego wymaga się dodatkowo aby zadania umieszczane w pojemniku tworzyły sekwencję:
$\forall_{i\in J_n}\forall_{j,k\,J\in D_j} j \in A(L^i)(s) \wedge l \in A(L^i)(s) \wedge j < k < l \Rightarrow k \in A(L^i)(s),\ dla\ s \in J_{|A(L^i)|}$

W niniejszej pracy przedstawiono przykład algorytmu sekwencyjnego $S1_k$ oraz przeprowadzono pełną analizę jego zachowania. Dowiedziono twierdzenia określającego wartość współczynnika sprawności algorytmu ($R_{S1_k} \geq 0.5$) dla najgorszego przypadku. Analiza probabilistyczna przeprowadzona została dla nieskończonego ciągu niezależnych zmiennych losowych o jednakowym rozkładzie $L = \{\xi_1, \xi_2, ...\}$. W pracy pokazano ograniczenie dla asymptotycznego współczynnika nadwyżki algorytmu $S1_k$ ($R^{\infty}_{S1_k, U(0,1)} > 0.7288$).

1 Sformułowanie problemu

Klasyczne zadanie pakowania polega na rozmieszczeniu listy ładunków $L = (a_1, \ldots, a_n)$, o rozmiarach nie przekraczających 1, w minimalnej ilości pojemników o rozmiarze jednostkowym. Wymaga się przy tym, aby żaden z pojemników nie był przeładowany, tj. aby suma elementów zapakowanych do niego nie przekraczała wartości 1. Tak postawiony problem posiada różnorakie praktyczne zastosowania poczynając od optymalnego załadowania ciężarówek czy zapełnienia telewizyjnych bloków reklamowych, aż po równomierne obciążanie pracą procesorów w maszynach wieloprocesorowych czy optymalizację przesyłania pakietów w sieciach komputerowych [1].

Połóżmy *I*=]0,1].

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

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

Element $L \in L$ nazywamy listą, jeżeli $L \in L_n$ 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.
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:

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

Liczbę *k* nazywamy mocą rozbicia P_k^B.
Oznaczmy przez P_k^B podzbiór zbioru $(2^B)^{Jk}$ składający się ze wszystkich k-elementowych rozbić zbioru B, zaś przez P^B zbiór $\bigcup_{k=1}^{\infty} P_k^B$.

Jeśli $p \in P_k^B$, to przez |p| oznaczamy moc p.
Niech dalej $P^{(n)}$ oznacza zbiór $\bigcup_{J \subset N, |J|=n} P^J$, zaś P zbiór $\bigcup_{i=1}^{\infty} P^{(n)}$.

DEFINICJA 3. Algorytmem pakowania A nazywamy odwzorowanie $A : L \to P$ spełniające następujące warunki:

$$\begin{aligned} &1.\ A(L) \in P^{D_L}; \\ &2.\ \forall_{j \in J_{|A(L)|}} \sum_{i \in A(L)(j)} L(i) \leq 1. \end{aligned} \tag{2}$$

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. Rozbicie p$\in P^{\mathrm{D_L}}$ nazywamy upakowaniem listy L, jeżeli

$$\forall_{i\in J_{|p|}} \sum_{i\in p(j)} L(i) \le 1 \tag{3}$$

Oznaczmy przez Pak(L) zbiór wszystkich możliwych upakowań listy L.

DEFINICJA 5. Niech L$\in$L. Upakowanie $\mathrm{OPT_L}$ spełniające warunek: $\forall_{p\in \mathrm{Pak(L)}} |p| \ge \mathrm{OPT_L}$ nazywamy optymalnym upakowaniem listy L.
Ze względu na skończoną długość listy L, a zatem również skończoną ilość rozbić zbioru $\mathrm{D_L}$, dla dowolnej listy istnieje upakowanie optymalne.

DEFINICJA 6. Niech OPT będzie takim algorytmem pakowania, że $\forall_{\mathrm{L}\in L} \mathrm{OPT(L)} = \mathrm{OPT_L}$. Wtedy algorytm OPT nazywamy optymalnym algorytmem pakowania.

DEFINICJA 7. Niech $\mathrm{A} \in A$. Stratą algorytmu pakowania A nazywamy różnicę

$$|\mathrm{A(L)}| - \sum_{i=1}^{|\mathrm{A(L)}|} \mathrm{lev(A(L))}(i)) \tag{4}$$

i oznaczamy ją $\mathrm{WS_A(L)}$.

Jedną z metod rozważania sprawności algorytmów jest metoda zwana analizą najgorszego przypadku. Polega ona na wyszukiwaniu takich list L, że algorytm daje najgorsze możliwe dla niego wyniki. Jeżeli potrafimy dowieść, że znaleziona lista jest poszukiwanym ”najgorszym przypadkiem”, to zyskujemy pewność, iż algorytm zachowuje się nie gorzej niż dla tej listy [2]. Formalnie analiza taka wymaga wprowadzenia kilku dalszych pojęć.

DEFINICJA 8. Niech A będzie algorytmem pakownia, L listą zadań. Współczynnikiem sprawności algorytmu A dla listy L nazywamy stosunek $\frac{|\mathrm{A(L)}|}{|\mathrm{OPT(L)}}$ i oznaczamy przez $\mathrm{R_A(L)}$.

Analogicznie definiujemy współczynnik sprawności dla dualnego algorytmu pakowania jako stosunek rezultatu tego algorytmu optymalnego.

DEFINICJA 9. Niech A będzie algorytmem pakowania.
Wyrażenie:

$$\max\{\mathrm{R_A(L)\ dla\ wszystkich\ list\ L,\ takich\ że\ OPT(L)} = n\} \tag{5}$$

nazywamy współczynnikiem nadwyżki algorytmu A i oznaczamy przez R_{A}^{n}.

DEFINICJA 10. Niech A będzie algorytmem pakowania. Wyrażenie: $\limsup_{n\to\infty} \mathrm{R}_{\mathrm{A}}^{n}$ nazywamy asymptotycznym współczynnikiem nadwyżki algorytmu A i oznaczamy przez R_{A}^{∞}.

W praktyce będziemy szukać asymptotycznego stosunku nadwyżki i tę liczbę uznawać za wartość sprawności algorytmu.

Podana wcześniej metoda nadania najgorszych przypadków daje co prawda pewność, że algorytm nie zachowa się gorzej niż przewidujemy, ale w wielu zastosowaniach przypadki najgorszych list albo wcale nie będą występować, albo będą bardzo rzadkie. Stąd pojawia się potrzeba znalezienia takiej miary sprawności algorytmu, która będzie lepiej oddawać rzeczywiste sytuacje. Poszukiwanie wyników działania algorytmu dla list generowanych losowo może lepiej pokazywać zachowanie się algorytmu. W badaniach takich standardowo zakłada się, że rozmiary poszczególnych elementów listy są wybierane niezależnie, zgodnie z jednakowym rozkładem prawdopodobieństwa, co poprawnie oddaje rzeczywiste warunki większości zadań. Metoda analizy probabilistycznej polega na znajdowaniu oczekiwanych wartości działania algorytmu dla list ładunków generowanych losowo. Rozkład z jakim generowane są ładunki ma oczywisty wpływ na wynik działania algorytmu, stąd właściwe dobieranie rozkładów w zależności od praktycznych potrzeb pozwala na dobór odpowiedniego algorytmu dla danego zadania [2].

Niech F będzie rozkładem prawdopodobieństwa dla zmiennych losowych $\mathrm{L}(i), i \in \mathrm{D_L}$, tj. dla ładunków. Rezultat algorytmu jest wtedy również zmienną losową. Analizę algorytmu A można w takim przypadku ograniczyć do badania wartości wyrażenia E[A(L)]. Dodatkowo, dla lepszego obrazu działania algorytmu, analizuje się jego oczekiwaną sprawność:

DEFINICJA 11. Niech A będzie algorytmem pakowania i niech $L_n = (\xi_1, \xi_2, ..., \xi_n)$, gdzie ξ_i są niezależnymi zmiennymi losowymi o jednakowym rozkładzie F.

Oczekiwaną sprawnością algorytmu A nazywamy $E[R_A(L_n)]$ i oznaczamy $R^n_{A,F}$.

DEFINICJA 12. Niech A będzie algorytmem dla zadania pakowania i niech $L_n = (\xi_1, \xi_2, ..., \xi_n)$, gdzie ξ_i są niezależnymi zmiennymi losowymi o jednakowym rozkładzie F. Wyrażenie :

$$\lim_{n \to \infty} \sup \; R^n_{A,F} \tag{6}$$

nazywamy asymptotyczną oczekiwaną sprawnością algorytmu A i oznaczamy przez $R^{\infty}_{A,F}$.

Przedstawimy teraz pewną wybraną klasę algorytmów dla takiego zadania – klasę algorytmów sekwencyjnych. Od algorytmu sekwencyjnego wymaga się dodatkowo aby zadania umieszczane w pojemniku tworzyły sekwencję:

DEFINICJA 12. Niech A będzie *n*-przebiegowym algorytmem pakowania. Jeżeli

$$\forall_{i \in J_n} \forall_{j,k,l \in D_i} j \in A(L^i)(s) \wedge l \in A(L^i)(s) \wedge j < k < l \Rightarrow k \in A(L^i)(s), \; dla\; s \in J_{\left|A(L^i)\right|} \tag{7}$$

to algorytm taki nazywamy sekwencyjnym algorytmem pakowania.

2 Algorytm sekwencyjny *S1$_k$*

Rozpatrzmy teraz następujący algorytm: w *i*-tym przebiegu (dla *i=k,k*-1,…1,0) badamy sumy kolejnych 2^i elementów z listy $L \mid (D_L \setminus \cup_{j \in J_{i-1}} D_j)$. Jeżeli suma tych ładunków nie przekracza 1 to pakujemy je do pojemnika, w przeciwnym przypadku przechodzimy do następnej grupy 2^k zadań i tak aż do wyczerpania listy. Zbiór D_i składa się ze wszystkich indeksów zadań zapakowanych według tej zasady w *i*-tym kroku. Na koniec pozostałe zadania pakujemy do nowego pojemnika.

3 Analiza najgorszego przypadku dla algorytmu $S1_k$

Dla tak określonego algorytmu można łatwo pokazać następujący lemat [3].

LEMAT 1. Dla algorytmu $S!_k$ i dla dowolnej listy L zachodzi:

$$WS_{S1_k}(L) < \left|S1_k(L)\right|. \tag{8}$$

Na podstawie powyższego lematu pokażemy ograniczenie dla najgorszego przypadku:

TWIERDZENIE 1.

$$R^{\infty}_{S1_k} = \frac{1}{2} \tag{9}$$

Dowód: Zauważmy najpierw, że $\left|OPT(L)\right| \le \left|S1_k(L)\right| + WS_{S1_k}(L) < 2\left|S1_k(L)\right|$ co daje $R^{\infty}_{S1_k} \ge \frac{1}{2}$. Z drugiej strony rozpatrując listę:

$l = \left\{ \underbrace{1-\frac{2}{n},\ldots,1-\frac{2}{n}}_{n\ razy}, \underbrace{\frac{1}{n},\ldots,\frac{1}{n}}_{2n\ razy} \right\}$ dostajemy |OPT(*L*)|=*n* oraz $\left|S1_k(L)\right| = \frac{1}{2} + 2$, a stąd $R_{S1_k} = \frac{\left|S1_k(L)\right|}{\left|OPT(L)\right|} = \frac{1}{2} + \frac{2}{n}$. Przechodząc do granicy dostajemy i uwzględniając $R^{\infty}_{S1_k} \ge \frac{1}{2}$ dostajemy tezę twierdzenia.

4 Analiza probabilistyczna algorytmu $S1_k$

Do analizy probabilistycznej omawianego algorytmu wykorzystanie model probabilistyczny opierający się na nieskończonych ciągach niezależnych zmiennych losowych o jednakowym rozkładzie [3].

Oznaczmy przez f_i i g_i gęstości rozkładów zmiennych $\xi^{(i)}$ i ξ^{i}, przez p_i i q_i prawdopodobieństwa wyboru i pozostawienia na liście ładunku $\xi^{(i)}$ w i-tym przebiegu algorytmu oraz przez Y_i i Z_i liczbę ładunków zapakowanych w i-tym przebiegu i liczbę pozostałych na liście $L^{(i)}$ ładunków po i-tym przebiegu.

W dalszych rozważaniach przyjmiemy dla zmiennych losowych rozkład jednostajny na przedziale [0,1].

Zauważmy, że f_i jest splotem g_{i-1} i g_{i-1}, czyli:

$$f_i(x) = \int_{-\infty}^{\infty} g_{i-1}(y) g_{i-1}(x-y)dy =$$
$$= 2^{2(i-1)} x^{2^i-1} \sum_{j=0}^{2^{i-1}-1} \binom{2^{i-1}-1}{j} \frac{(-1)^j}{2^{i-1}+j} \tag{10}$$

Stąd dostajemy:

$$p_i = 1 - q_i \qquad g_i = \frac{f_i(x)}{q_i} = 2^i x^{2^i-1} \tag{11}$$

Rozkłady zmiennych losowych Y_i zbadamy korzystając ze schematu Bernoulliego z prawdopodobieństwem porażki q_1=0.5:

$P(Y_i = j | Z_{i-1} = 2m) = \binom{m}{j} p_i^j q_i^{m-j}$. Korzystając teraz ze wzoru na prawdopodobieństwo całkowite dostajemy:

$$P(Y_i = j) = \sum_{m=0}^{n} \binom{m}{j} p_i^j q_j^{m-j} (P(Z_{i-1} = 2m) + P(Z_{i-1} = 2m+1))$$
$$P(Z_i = j) = \sum_{m=0}^{n} \binom{m+j}{j} p_i^m q_j^j (P(Z_{i-1} = 2j+2m) + P(Z_{i-1} = 2j+2m+1)) \tag{12}$$

Korzystając z własności warunkowej wartości oczekiwanej prowadzi to do nierówności:

$$\frac{p_i}{2} E(Z_{i-1}) - \frac{p_i}{2} \le E(Y_i) \le \frac{p_i}{2} E(Z_{i-1}) \tag{13}$$

A to pozwala nam już oszacować wartość oczekiwaną rezultatu algorytmu $S1_k$:

$$E(|S1_k(L_n)|) \ge \sum_{i=1}^{k} (\frac{p_i}{2} E(Z_i) - \frac{p_i}{2}) \ge \sum_{i=1}^{k} \frac{E(Z_i)}{2} - \frac{E(Z_k)}{2} - \frac{k}{2} \tag{14}$$

i analogicznie od góry

$$E(|S1_k(L_n)|) = \sum_{i=1}^{k} E(Y_i) \le \frac{E(Z_0)}{2} - \sum_{i=1}^{k} \frac{E(Z_i)}{2} - \frac{E(Z_k)}{2} + k \tag{15}$$

Ponieważ wartości q_i szybko zbiegają do 0:

q_1=0.5,
q_2=0.1667,
q_3=0.0143,
q_4=7.771·10^{-4},
q_5=1.663·10^{-9},

$q_6=5.457\cdot10^{-19}$,
$q_7=4.175\cdot10^{-38}$,
…

więc również wartości $E(Z_i)$ szybko maleją. Zatem w powyższych wzorach istotne znaczenie mają jedynie pierwsze składniki sumy. Poniżej przedstawiono oszacowanie oczekiwanego rezultatu algorytmu $S1_k$ korzystające z pierwszych trzech składników odpowiednich sum. Daje to ograniczenie od góry:

$$E(|S1_k(L_n)|) > \frac{E(Z_0)}{2} - \frac{E(Z_1)}{2} - \frac{E(Z_2)}{2} - \frac{2E(Z_3)}{2} - k - 1 \tag{16}$$

oraz z dołu:

$$E(|S1_k(L_n)|) < \frac{E(Z_0)}{2} - \frac{E(Z_1)}{2} - \frac{E(Z_2)}{2} - \frac{E(Z_3)}{2} + k \tag{17}$$

Pozostaje jeszcze oszacowanie wartości $E(Z_0)$, $E(Z_1)$, $E(Z_2)$ i $E(Z_3)$. Z określenia Z_i i wcześniejszych zależności mamy: $E(Z_0)=n$

$$\begin{aligned} &\frac{n}{4} \le E(Z_1) \le \frac{n}{4}+1 \\ &\frac{n}{48} \le E(Z_2) \le \frac{n}{48}+\frac{13}{12} \\ &\frac{n}{6720} \le E(Z_3) \le \frac{n}{6720}+\frac{1693}{1680} \end{aligned} \tag{18}$$

Ostatecznie otrzymujemy następujące oszacowanie badanej wartości oczekiwanej:

$$\frac{2449}{6720}n - k - \frac{142}{35} \le E(|S1_k(L_n)|) \le \frac{1633}{4480}n + k \tag{19}$$

Ze względu na określenie algorytmu $S1_k$ mamy $k \le \log_2 n$, więc:

$$0.364434n - \log_2 n - 5 < E(|S1_k(L_n)|) < 0.364505n + \log_2 n \tag{20}$$

Daje to bardzo dokładne oszacowanie asymptotycznego zachowania się wartości oczekiwanej wyniku algorytmu, czyli liczby zapełnionych pojemników.

Na koniec podamy jeszcze oszacowanie dla oczekiwanej sprawności algorytmu $S1_k$. Korzystając z oszacowania dla wartości oczekiwanej algorytmu optymalnego uzyskanego przez J. Cirika, J. Frenka, G. Galambosa i A. Rinoy Kana ([4]):

$$E(|OPT(L_n)|) \le \frac{1}{2}\left(1-\frac{1}{2^n}\binom{2}{n/2}\right) \tag{21}$$

Dostajemy:

$$R^n_{S1_k,U(0,1)} > \frac{\frac{2449}{6720}n-k-\frac{142}{35}}{\frac{n}{2}\left(1-\frac{1}{2^n}\binom{n}{n/2}\right)} > \frac{0.7288n-2(k+5)}{n\left(1-\frac{1}{2^n}\binom{n}{n/2}\right)} \tag{22}$$

Zauważmy, że:

$$\lim_{n\to\infty} 2^{-n}\binom{n}{n/2} = 0 \tag{23}$$

Dostajemy zatem ograniczenie dolne dla asymptotycznej sprawności algorytmu $S1_k$:

$$R^{\infty}_{S1_k,U(0,1)} > 0.7288 \tag{24}$$

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] Horzelski W.: *Algorytmy sekwencyjne dla zadania pakowania*, Rozprawa doktorska obroniona na Wydziale Matematyki UŁ

[4] Cirik J., Frenk J., Galambos G. i Rinoy Kan A.: *Probabilistic Analysis of Algorithms for Bin Packing Problems*, Journal of Algorithms 12 (1991), 189-203

PROBABILISTIC ANALYSIS OF SOME SEQUENTIAL ALGORITHMS FOR BIN-PACKING PROBLEM

Summary – The bin-packing problem in the classical approach is to arrange the list of tasks L={a1,a2,…,an} of a size not exceeding 1 in the minimum number of bins of size 1, however, that none of the bins was overloaded. Among the ways to do such a task is a class of sequential algorithms. From sequential algorithm is required in addition to pack the

tasks in such a way that tasks placed in each container (bin) consisted of a sequence:

$$\forall_{i\in J_n}\forall_{j,k,l\in D_i} j \in A(L^i)(s) \wedge l \in A(L^i)(s) \wedge j < k < l \Rightarrow k \in A(L^i)(s),\ dla\ s \in J_{\left|A(L^i)\right|}$$

In this paper is an example of sequential algorithm called $S1_k$ and carried out a full analysis of its behavior. It demonstrate the value of lower bound for efficiency factor of the algorithm ($R_{S1_k} \geq 0.5$). Probability analysis was carried out for the infinite sequence of independent random variables of equal distribution $L = \{\xi_1, \xi_2, ...\}$. This paper also show a bound of asymptotic waste ratio of $S1_k$ ($R^{\infty}_{S1_k, U(0,1)} > 0.7288$).