ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1992
Seria: AUTOMATYKA z. 109 Nr kol. 1175

Jolanta Sobczyńska

Instytut Badań Systemowych PAN

O EFEKTYWNOŚCI GENERATORÓW PSEUDOLOSOWYCH DLA POTRZEB OPTYMALIZACJI ZERO-JEDYNKOWEJ

ON EFFICIENCY OF PSEUDO-RANDOM GENERATORS FOR ZERO-ONE OPTIMIZATION PROBLEMS

ОБ ЭФФЕКТИВНОСТИ ГЕНЕРАТОРОВ СЛУЧАЙНЫХ ЧИСЕЛ ДЛЯ ЗАДАЧ ОПТИМИЗАЦИИ С БУЛЕВЫМИ ПЕРЕМЕННЫМИ

Streszczenie: W pracy zostały przedstawione dwa sposoby generowania n-wymiarowych wektorów binarnych. Omówiono cechy tych generatorów pod kątem ich zastosowania do rozwiązywania wielowymiarowego zagadnienia załadunku. Opisano też implementację równoległą procesu generowania ciągu wektorów losowych na karcie 4-transputerowej.

Summary : Two ways of generating 0-1 n-dimensional random vectors are described. Usefulness of these generators for solving multi-knapsack problem is discussed. A parallel implementation of the vector generating process on transputers is presented.

Резуме : Описаны два способа генерирования случайных двоичных векторов длины n. Обсуждено применение этих подходов к многомерной задаче о ранце. Представлены результаты параллельной имплементации процесса генерирования последовательностей случайных векторов на транспьютерной плате.

1. Wstęp

Problem generowania ciągów liczb losowych występuje w wielu grupach zagadnień. Jedną z takich grup stanowią zadania numeryczne rozwiązywane metodami Monte-Carlo. Zadanie numeryczne jest zastępowane zadaniem z rachunku prawdopodobieństwa tak, że znany jest związek między rozwiązaniami obu zadań, a zadanie z rachunku prawdopodobieństwa rozwiązujemy na drodze eksperymentu statystycznego (na ogół jest to losowanie próbki z odpowiedniej populacji, czyli generowanie ciągu liczb losowych) [7]. Do tej grupy należą również zadania optymalizacji dyskretnej, rozwiązywane metodami poszukiwania losowego. Podczas tego procesu problem generowania wektorów jest jednym z kluczowych.

W pracy zostaną przedstawione dwa sposoby generowania n-wymiarowych wektorów 0-1 dla zadań optymalizacji dyskretnej, a przede wszystkim dla wielowymiarowego zagadnienia załadunku. Efektywna metoda generowania wektorów pseudolosowych jest szczególnie ważna w zadaniach o złożonej strukturze ograniczeń. Dobranie wówczas generatora o akceptowalnej sprawności (stosunek liczby wektorów dopuszczalnych do liczebności próby) ma decydujący wpływ zarówno na czas poszukiwań, jak i na dokładność rozwiązania.

Oba przedstawione generatory wektorów 0-1 oparte są na tym samym generatorze liczb pseudolosowych z przedziału (0,1). Klasa generatorów multiplikatywnych, do której on należy, oraz sam generator zostały zaprezentowane w rozdziale 2. W rozdziale 3 sformułowano problem (wielowymiarowe zagadnienie załadunku - WZZ), do rozwiązania którego generatory zostały użyte. Podstawową część pracy stanowi rozdział 4, w którym zaprezentowano oba generatory, ich cechy, i dokonano krótkiego porównania ze względu na ich przydatność dla rozwiązywania zadań WZZ. Rozdział 5 zawiera krótki opis i wyniki równoległej implementacji procesu generowania wektorów na karcie 4-transputerowej.

2. Klasa generatorów multiplikatywnych i generator liczb rzeczywistych RAND

Do klasy generatorów multiplikatywnych należą generatory postaci

$$x_{n+1} = c \cdot x_n \pmod M$$

przy czym c, x_0 (a więc również wszystkie x_n) są liczbami całkowitymi z przedziału [0,M). Wtedy liczby $r_n = x_n/M$ są liczbami rzeczywistymi z przedziału [0,1). Szczegóły dotyczące cech generatorów z tej klasy można znaleźć w literaturze, m. in. [6].

Do klasy generatorów multiplikatywnych należy generator liczb pseudolosowych RAND [4]. Służy on do generowania losowych liczb rzeczywistych z przedziału (0,1). W generatorze RAND stałe mają następujące wartości: $c = 7^5 = 16807$, $M = 2^{31} - 1 = 2147483647$ (liczba pierwsza Mersenne'a). Wszystkie otrzymane liczby x_i spełniają warunek : $0 < x_i < 2^{31} - 1$ (i są całkowite). Generator RAND jest napisany jako funkcja. Jego kod fortranowy można znaleźć w [4]. Tam też znajdują się informacje na temat cykliczności i właściwości statystycznych tego generatora. Generator RAND zostanie przez nas użyty do generowania ciągów wektorów zero-jedynkowych.

3. Sformułowanie problemu WZZ

W [1] przedstawiono algorytmy rozwiązywania zadania optymalizacji dyskretnej ze zmiennymi binarnymi, które zapewniają losowy lub deterministyczny przedział ufności na zadanym poziomie ufności. O efektywności czasowej tych algorytmów decyduje generator liczb pseudolosowych wykorzystywany do tworzenia losowych wektorów zero-jedynkowych. Algorytmy z probabilistyczną oceną dokładności mogą być przydatne do rozwiązywania trudnych zadań optymalizacji dyskretnej. Przykładem takiego zadania jest binarne wielowymiarowe zadanie załadunku. Zadanie to ma następującą postać :

$$\max z = \sum_{j=1}^{n} c_j x_j$$

$$\sum_{j=1}^{n} a_{ij} x_j \leqslant b_i \ , \ i = 1,\ldots,m,$$

$$a_{ij} \geqslant 0,\ b_i > 0,\ c_j > 0,\ x_j = 0 \text{ lub } 1,\ i = 1,\ldots,m;\ j = 1,\ldots,n,$$

a_{ij} , b_i , c_j - liczby rzeczywiste.

4. Dwa sposoby generowania pseudolosowych n-wymiarowych wektorów zero-jedynkowych

Biorąc generator liczb rzeczywistych RAND jako bazowy, skonstruowano dwa generatory wektorów 0-1 o zadanej długości n. Drugi sposób generacji jest wynikiem doświadczeń uzyskanych w pracy z pierwszym generatorem, w zastosowaniu do rozwiązywanego zadania WZZ (patrz rozdział 3)

a) Generator 2NRAND

Generator 2NRAND generuje w sposób niezależny wektory z przestrzeni 2^n (n-długość wektora), czyli z przestrzeni wszystkich wektorów 0-1 o długości n. Schemat tego generatora jest następujący :

1) generacja liczby rzeczywistej $z \in (0,1)$ (generator RAND),
2) przekształcenie liczby z w liczbę całkowitą iz równą 0 lub 1 :

$$iz = 2 \cdot z \; ; \qquad /1/$$

otrzymujemy to przekształcenie, korzystając z ogólnego wzoru:

$$iz = (J-I+1) \cdot z + I \; , \qquad /2/$$

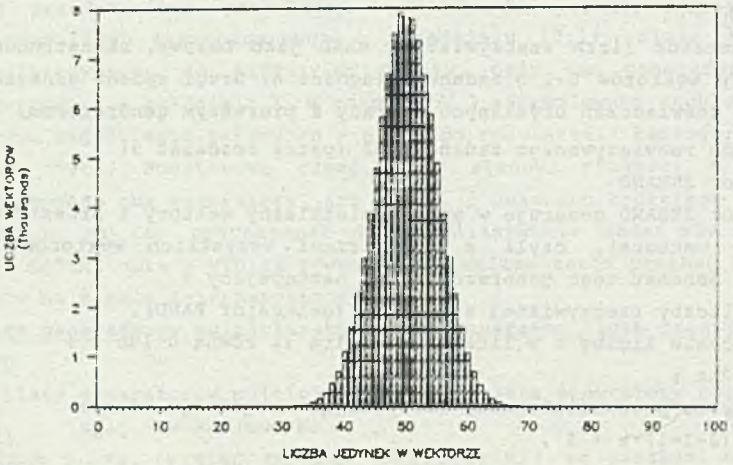
które zamienia liczbę rzeczywistą z przedziału (0,1) na liczbę całkowitą z przedziału (I,J). Ustalając I=0, J=1, otrzymujemy /1/,

3) wstawienie liczby iz na miejsce kolejnej współrzędnej w generowanym wektorze.

Ten sposób generacji wektorów 0-1 powoduje, że nie mamy żadnej kontroli nad podobszarem generowania wektorów w obszarze wszystkich wektorów (czyli nad liczba i rozkładem jedynek w generowanym wektorze). W przypadku gdy nie posiadamy żadnych informacji o rozkładzie wektorów dopuszczalnych (czyli spełniających ograniczenia z rozdziału 3) wśród wszystkich wektorów, nie jest to wadą. Jeśli jednak mamy pewną wiedzę na ten temat (np. wiemy, że rozkład jest skupiony w jakimś podobszarze, albo znamy granicę górną i dolną liczby jedynek dla wektorów optymalnych), nie mamy możliwości wykorzystania tych informacji podczas generowania wektorów za pomocą generatora 2NRAND.

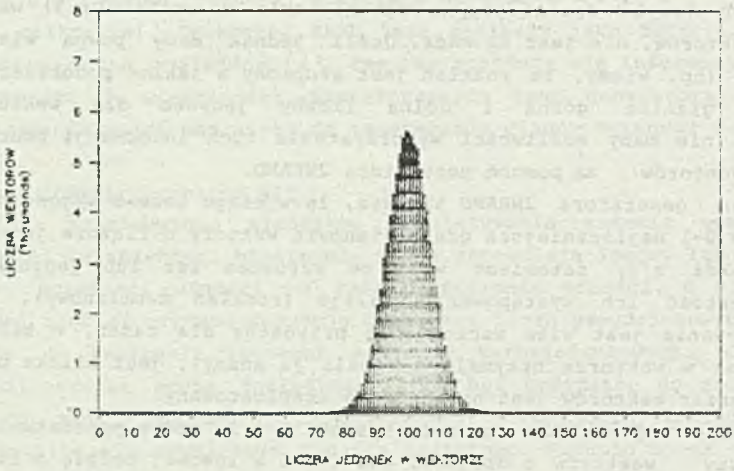
Konstrukcja generatora 2NRAND sprawia, że w ciągu losowo wygenerowanych wektorów 0-1 najliczniejszą grupę stanowią wektory o liczbie jedynek skupionej wokół n/2, natomiast wraz ze wzrostem zer lub jedynek w wektorze częstość ich występowania maleje (rozkład dwumianowy). Ten rodzaj generowania jest więc szczególnie przydatny dla zadań, w których liczba jedynek w wektorze optymalnym (jeśli ją znamy), jest bliska n/2, bowiem ten obszar wektorów jest najbardziej eksploatowany.

Potwierdzenie tej analizy stanowią rysunki 1 i 2, które przedstawiają rozkład liczby wektorów o długości 100 i 200 w losowej próbie o liczności 100 000, bez ograniczeń nakładanych na generowane wektory.



Rys. 1. Generowanie 100000 wektorów o długości 100 przy użyciu generatora 2NRAND

Fig. 1. Generating of 100000 vectors by 2NRAND (length=100)



Rys. 2. Generowanie 100000 wektorów o długości 200 przy użyciu generatora 2NRAND

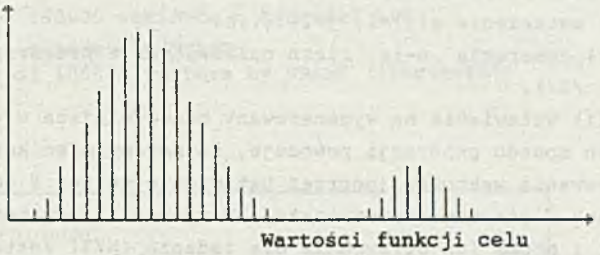
Fig. 2. Generating of 100000 vectors by 2NRAND (length=200)

Rysunki 1 i 2 wykazują symetrię rozkładu jedynek względem osi x=n/2. Taka sytuacja ma miejsce, gdy w procesie generowania wektorów nie ingerujemy z ograniczeniami wynikającymi ze struktury zadania.

Jeśli proces generowania wektorów 0-1 zastosujemy w celu rozwiązania konkretnego zadania (WZZ) z ograniczeniami, wówczas rozkład częstości jedynek ulega deformacji. Część wektorów z prawej strony rozkładu zostaje odcięta, a ponadto pewna liczba wektorów z centralnej masy rozkładu okazuje się również niedopuszczalna i jest eliminowana.

Metoda rozwiązywania zadania (WZZ) przy użyciu poszukiwania losowego polega na generowaniu ciągu wartości dopuszczalnych funkcji celu (wartości funkcji celu realizowanych przez wektory dopuszczalne) do oceny jakości rozwiązania metodą losowych lub deterministycznych przedziałów o zadanym poziomie ufności [1].

Specyficzna struktura zadania (WZZ) może spowodować czasami, że nałożenie ograniczeń na generowane wektory powoduje powstanie w rozkładzie częstości wartości funkcji celu obszaru/punktu izolowanego (częstość punktów tego obszaru nie musi być jednakowa, może być duża (patrz rys. 3).



Rys.3. Istnienie obszaru/punktu izolowanego w rozkładzie częstości wartości funkcji celu

Fig.3. The occurrence of isolated region/point in goal function frequency distribution

W takiej sytuacji wygenerowanie odpowiednio długiego ciągu dopuszczalnych wartości funkcji celu z centralnego obszaru rozkładu (stanowiącego prawie całą jego masę) może zapewnić żądaną jakość probabilistyczną uzyskanego rozwiązania, lecz odstęp między wartością optymalną i rozwiązaniem jest duży (20-49%) [1]. Stąd podstawowym zadaniem do rozwiązania w przyszłości staje się umiejętność rozpoznawania zadań tego typu na podstawie ich opisu (wektor współczynników funkcji celu - c, macierz ograniczeń - A, wektor prawych stron ograniczeń - b, oraz ich analiza). Pomocna może być również próba złożenia rozkładu otrzymanego z 2NRAND, z innym rozkładem otrzymanym przez jakiś generator wektorów tak, aby w rezultacie otrzymać równomierny rozkład generowanych wektorów.

Pewną przeciwwagę dla generatora 2NRAND stanowi drugi z prezentowanych generatorów - NRAND.

b) Generator NRAND

Generator NRAND generuje niezależnie i w sposób równomierny wektory o różnej liczbie jedynek (tzn. wektor z dowolną liczbą jedynek jest jednakowo prawdopodobny). Schemat generatora NRAND jest dwuetapowy.

ETAP 1. Generacja liczby jedynek w wektorze:

1) generacja liczby rzeczywistej z z przedziału (0,1) (generator RAND),
2) przekształcenie liczby z w liczbę całkowitą iz z przedziału [0,n] - wykorzystanie wzoru /2/; liczba iz jest liczbą jedynek w generowanym wektorze.

ETAP 2. Wypełnianie wektora:

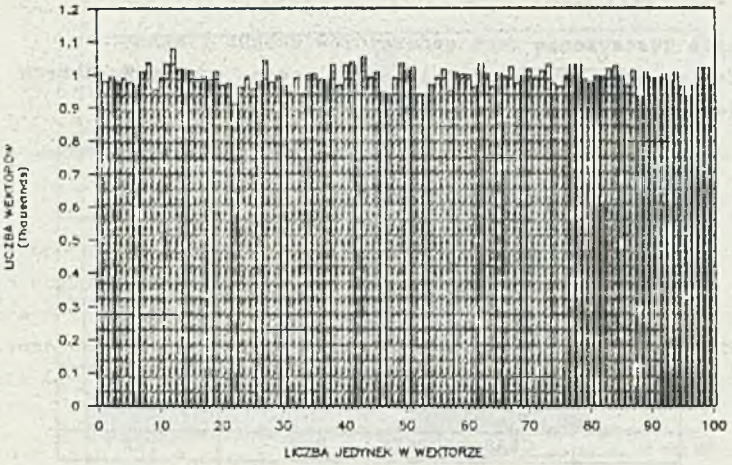
1) jeśli iz < n/2, to
   i) generacja iz liczb całkowitych z przedziału [1,n] (stosując wzór /2/),
   ii) wstawienie na wygenerowany numer miejsca w wektorze liczby 1;
2) jeśli iz > n/2, to
   i) ustawienie x(j)=1, j=1,...,n,
   ii) generacja n-iz liczb całkowitych z przedziału [1,n] (stosując /2/),
   iii) wstawienie na wygenerowany numer miejsca w wektorze liczby 0.

Ten sposób generacji powoduje, że możemy mieć kontrolę nad obszarem generowania wektorów (poprzez ustawienie dolnej i górnej granicy liczby jedynek w generowanym wektorze). W sytuacji gdy potrafimy podać te wartości (próba ich otrzymania dla zadania (WZZ) została zrobiona w [2]), obszar poszukiwań zostaje zawężony do obszaru najbardziej dla nas istotnego.

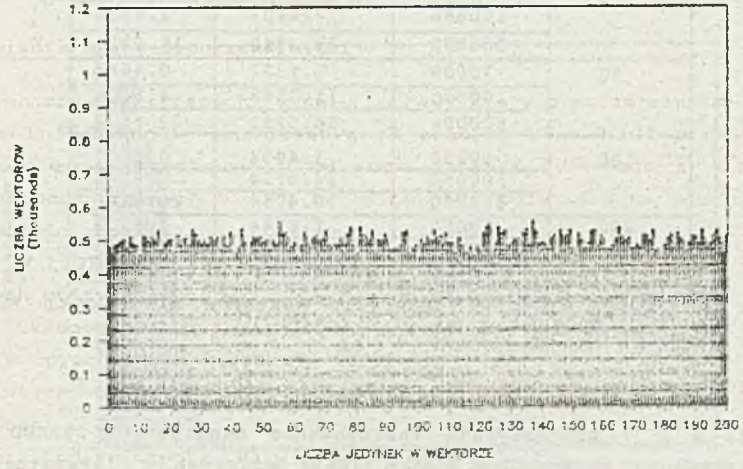
Schemat generatora NRAND pokazuje, że rozkład częstości wektorów o różnej liczbie jedynek jest równomierny w przedziale [0,n]. Ilustrują to rysunki 4 i 5 (patrz następna strona), które przedstawiają rozkład z próbki losowej wektorów o długości 100 i 200 w losowej próbie o liczności 100 000, bez ograniczeń. Pokazują one (podobnie jak w przypadku generatora 2NRAND, ale w inny sposób) symetrię rozkładu jedynek wokół osi x=n/2. Ponieważ jednak różnych wektorów o małej liczbie jedynek (lub zer) jest niewiele (w porównaniu z liczebnością zbioru wektorów o liczbie jedynek bliskiej połowie), częściej generowane są te same wektory z serii wektorów o małej liczbie jedynek lub zer.

Zastosowanie procesu generowania wektorów do konkretnego zadania (WZZ) z ograniczeniami powoduje, naturalnie, eliminację części wektorów, jako niedopuszczalnych.

Jeśli zbiór wektorów dopuszczalnych jest rzadki (albo wartość optymalna jest w podobszarze/punkcie izolowanym), to nawet znajomość granic (górnej i dolnej) liczby jedynek w wektorze optymalnym może okazać



Rys. 4. Generowanie 100000 wektorów o długości 100
przy użyciu generatora NRAND
Fig. 4. Generating of 100000 vectors by NRAND (length=100)



Rys. 5. Generowanie 100000 wektorów o długości 200
przy użyciu generatora NRAND
Fig. 5. Generating of 100000 vectors by NRAND (length=200)

się niewystarczająca do efektywnego generowania wskazanego obszaru wektorów.

c) Krótkie zestawienie cech generatorów 2NRAND i NRAND

Zestawienie kilku najważniejszych cech obu generatorów zostało przedstawione w Tabeli 1.

Lp.	CECHA	2NRAND	NRAND
1.	NIEZALEŻNOŚĆ WEKTORÓW	TAK	NIE
	RÓWNOMIERNOŚĆ ROZKŁADU WEKTORÓW	NIE	TAK
	KONTROLA NAD OBSZAREM GENEROWANIA	NIE	TAK
	CZY ANLIZA ZADANIA MOŻE POMÓC	?	TAK
	CZAS KRÓTSZY	NIE	TAK

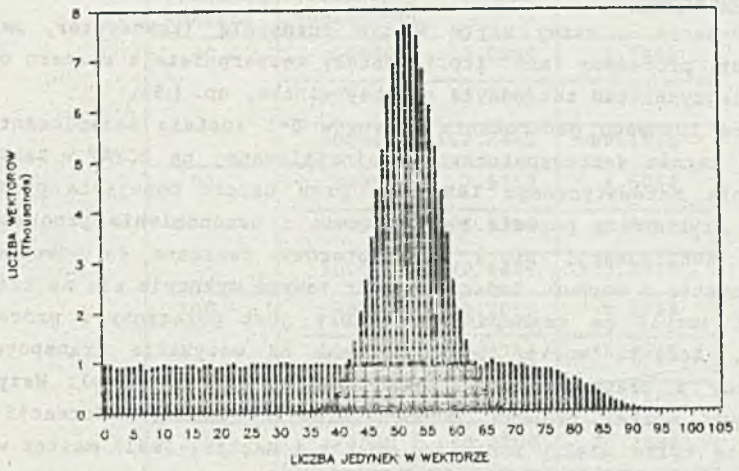
Tabela 1. Zestawienie niektórych cech generatorów 2NRAND i NRAND

Tabela 2 przedstawia zestawienie czasowe (w sekundach) generowania próbki wektorów różnej liczności, o różnej długości. Czasy te zostały uzyskane na IBM 3090 VF.

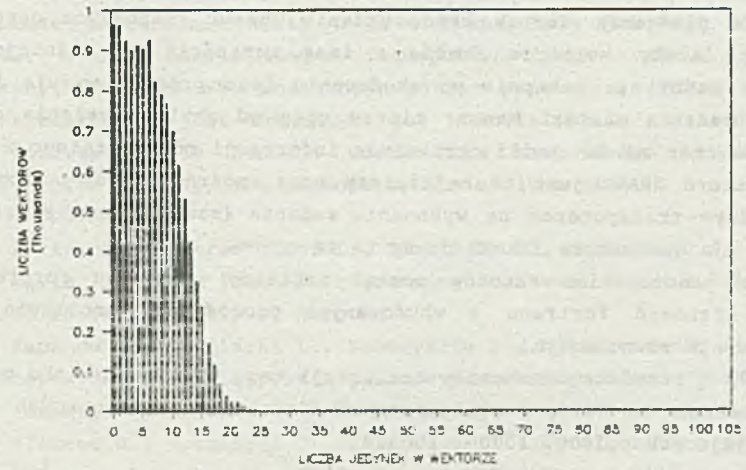
długość wektora	liczba wektorów	2NRAND	NRAND
20	10000	0.7308	0.2992
	100000	7.4403	3.0836
	500000	37.2960	15.5381
50	10000	1.1382	0.4643
	100000	11.3568	4.5825
	500000	56.5992	22.8410
100	10000	1.4094	0.5687
	100000	14.0724	5.6348
	500000	70.4754	28.2185
200	10000	1.6544	0.6728
	100000	16.5632	6.7349
	500000	82.7024	33.6298

Tabela 2. Wyniki czasowe generacji wektorów dla próbki losowej o ustalonej długości na IBM 3090 VF

Rysunki 6 i 7 przedstawiają zastosowanie generatorów 2NRAND i NRAND do rozwiązywania konkretnych zadań (WZZ) wziętych z literatury [3]. Pokazują one, jak bardzo sposób generowania wektorów wpływa na charakter zachowania się krzywej obrazującej rozkład liczby wartości funkcji celu zintegrowanych do przedziałów. Rysunek 6 został sporządzony dla przykładu WEIN7, rysunek 8 - dla przykładu WEIN8. Oba przykłady mają ten sam wektor współczynników funkcji celu, tę samą macierz ograniczeń.



Rys. 6. Generatory 2NRAND i NRAND (100000 wekt.) dla WEIN7 (n=105,m=2)
Fig. 6. Generators 2NRAND and NRAND (100000 vect.) for WEIN7 (n=105,m=2)



Rys. 7. Generatory 2NRAND i NRAND (100000 wekt.) dla WEIN8 (n=105,m=2)
Fig. 7. Generators 2NRAND and NRAND (100000 vect.) for WEIN8 (n=105,m=2)

5. Równoległa implementacja procesu generowania wektorów 0-1 na karcie 4-transputerowej

Uwaga: Pojęcia i nazwy użyte w tym rozdziale (transputer, master, worker, root, processor farm itp.) zostały zaczerpnięte z obszaru obliczeń równoległych, tam też odsyła się czytelnika, np. [5].

Procedura losowego generowania wektorów 0-1 została zaimplementowana również na karcie 4-transputerowej, zainstalowanej na PC/AT w Zakładzie Programowania Matematycznego IBS PAN, przy użyciu podejścia processor farm. Ten tryb pracy pozwala na napisanie i uruchomienie programu bez znajomości konfiguracji sieci transputerów. Tworzone są wówczas dwa zadania : master i worker. Zadanie master zawsze wykonuje się na transputerze root (czyli na transputerze, który jest połączony z procesorem komputera). Zadanie worker jest ładowane na wszystkie transputery w sieci (każdy z transputerów ma załadowane to samo zadanie). Wszystkie zadania typu worker pracują jednocześnie, a wymiana informacji może mieć miejsce tylko między zadaniami worker i master. Jeśli master wysyła informację, jest ona przejmowana przez worker, który aktualnie jest wolny. To podejście jest bardzo przydatne w przypadku, gdy chcemy rozwiązać kilka zadań tego samego typu. Tak właśnie jest w przypadku generacji wektorów 0-1. Praca zorganizowana jest w prosty sposób. Master otrzymuje od użytkownika następujące informacje : liczba wektorów do generowania, długość generowanych wektorów oraz wartości zmiennych, inicjujących działanie funkcji RAND w każdym z transputerów. Następnie master oblicza, ile wektorów musi wygenerować każdy transputer (dzieli pracę między 3 transputery) i przesyła potrzebne dane do każdego transputera.
W momencie przejęcia danych każde zadanie worker rozpoczyna generację określonej liczby wektorów (każde z inną wartością do zainicjowania generatora RAND), a następnie po skończeniu tego procesu wysyła informację do zadania master. Master mierzy czas od chwili wysłania danych do zadań worker aż do chwili otrzymania informacji od ostatniego z nich. Dla generatora NRAND jest to najdłuższy czas spośród czasów potrzebnych poszczególnym transputerom na wykonanie zadania (czasy te nie są równe). Natomiast dla generatora 2NRAND czasy te są równe.

Program generowania wektorów został napisany w języku Fortran 3L. Jest to standard Fortranu z wbudowanymi procedurami umożliwiającymi pracę w trybie równoległym.

W Tabeli 3 przedstawiono czasy realizacji tego procesu dla obu generatorów w sekundach. Testy zostały przeprowadzone dla następujących zmiennych inicjujących : 1000, 10000, 100000.

długość wektora	liczba wektorów	2NRAND	NRAND
20	10000	3.0584	1.7846
	100000	30.5705	17.8031
	500000	152.8462	89.1858
50	10000	7.5752	4.6034
	100000	75.7302	45.6869
	500000	378.6404	228.3478
100	10000	15.1015	9.6418
	100000	148.9794	96.0263
	500000	754.8348	479.4601
200	10000	30.1509	20.2934
	100000	301.4473	202.3803
	500000	1507.0866	1011.4126

Tabela 3. Wyniki czasowe generowania próbki losowej wektorów o zadanej długości na karcie 4-transputerowej

Tabele 1 i 3 pokazują, że czas realizacji procesu generacji na IBM 3090 VF jest około 10-krotnie krótszy niż na karcie 4-transputerowej. Jednakże poszerzenie karty o nowe transputery powoduje skrócenie tego czasu, a cena zakupu nowych transputerów jest o wiele niższa od ceny maszyny IBM 3090 VF. Dodatkowym plusem przemawiającym na korzyść sieci transputerowej jest możliwość jej zainstalowania na komputerze klasy PC, dostępnym dla każdego użytkownika komputerów, podczas gdy zakup maszyny IBM 3090 VF jest niemożliwy dla wielu z nich.

LITERATURA

[1] Bertocchi M., Brandolini L., Słomiński L., Sobczyńska J.: A Monte Carlo Approach for 0-1 Programming Problems, Q.D.M.S.I.A., 1990, No 18., University of Bergamo.

[2] Kabanow P., Słomiński L., Sobczyńska J.: Wykorzystanie sieci transputerowych do rozwiązywania zadania plecakowego metodą poszukiwania losowego. Warszawa 1992 (w realizacji).

[3] Plateau G., Roucairol C., Valabregue I.: Un algorithme parallel PR^2 du multiknapsack, INRIA Research Report, n. 811, March 1988.

[4] Schrage L.: A More Portable Fortran Random Number Generator. ACM Transations on Mathematical Software, Vol. 5, No. 2, June 1979, pp. 132-138.

[5] Transputery - dokumentacja, INMOS Ltd.

[6] Zieliński R.: Generatory liczb losowych. WNT, Warszawa 1972.
[7] Zieliński R.: Metody Monte-Carlo. WNT, Warszawa 1970.

Recenzent: Doc.dr h.inż. Jan Kałuski
Wpłynęło do Redakcji do 30.04.1992 r.

Abstract:

The aim of the paper is to describe two ways of generating n-dimensional zero-one vectors for discrete optimization problems. Both of them use the same generator of pseudo-random real numbers between 0 and 1. The process of the vector generating is one of the most important steps in methods solving optimization problems by random search. For this process the time performance plays important role due to the large number of vector generating routine calls. The efficient method of vector generating is important in problems with complicated constraints structure. Choice of the generator with acceptable efficiency (it is the number of feasible vectors related to the sample cardinality) influences on the searching time and the solution accuracy).

Usefulness of both generators for the binary multiknapsack problem was tested on large set of test problems known in the literature. Results for some of them are shown in the paper.

At the end of the paper, the parallel implementation of the vector generating process on 4-transputer board is briefly described.