STUDIA INFORMATICA 2004
Volume 25 Number 2 (58)

Marcin WOCH
Politechnika Śląska, Instytut Informatyki

ROZWIĄZANIE PROBLEMU DOSTAW Z OKNAMI CZASOWYMI ZA POMOCĄ SYMULOWANEGO WYŻARZANIA

Streszczenie. Artykuł prezentuje nową wersję funkcji przejścia w algorytmie symulowanego wyżarzania. Celem tej metody jest znalezienie jak najlepszych rozwiązań problemu dostawy z oknami czasowymi. Algorytm opisany w niniejszym artykule jest implementacją sekwencyjnego symulowanego wyżarzania.

Słowa kluczowe: symulowane wyżarzanie, problem dostaw z oknami czasowymi

SOLVING VEHICLE ROUTING PROBLEM WITH TIME WINDOWS USING SIMULATED ANNEALING

Summary. Article describes a new version of function generating solutions in simulated annealing algorithm for vehicle routing problem with time windows. A goal of this method is to find solutions being a good approximation of the set of efficient solutions in the short time. A method used in this paper is a modification of sequentional simulated annealing algorithm.

Keywords: simulated annealing, vehicle routing problem with time windows

1. Symulowane wyżarzanie

Algorytm symulowanego wyżarzania jest rozwinięciem metod przeszukiwania lokalnego, które polegają na ulepszaniu istniejącego rozwiązania do momentu, gdy nie udaje się go dalej poprawić.

Podstawowy algorytm przeszukiwania lokalnego dla minimalizacji funkcji F wygląda następująco:

```
Krok 1. Stwórz rozwiązanie początkowe x0
Krok 2. Dopóki F(x0) < F(x)
```

```
Krok 2.1. Wybierz rozwiązanie x z sąsiedztwa N
Krok 2.2. Jeżeli F(x) < F(x0)
   x0 = x
```

Podstawową wadą powyższego algorytmu jest możliwość utknięcia w minimum lokalnym optymalizowanej funkcji. Symulowane wyżarzanie, pomimo znalezienia minimum lokalnego, pozwala na dalsze przejście „w górę", a tym samym na bardziej efektywne poszukiwanie optimum globalnego.

Podstawy symulowanego wyżarzania zostały po raz pierwszy opisane w roku 1953 przez Metropolis i in. [6] w algorytmie symulującym chłodzenie ciał stałych. W procesach ochładzania, jak i stygnięcia metali zaobserwowano, że jego właściwości będą zależały od szybkości chłodzenia. Przy powolnym stygnięciu, cząsteczki ciała oddając energię rozkładają się w sposób bardziej systematyczny tworząc równomierne struktury. Natomiast jeśli chłodzenie będzie zbyt szybkie, to cząsteczki rozłożą się bardziej chaotycznie.

Algorytm Metropolis, wzorujący się na prawach termodynamiki, symuluje zmianę energii systemu podczas procesu wyżarzania. Jeżeli energia spada, to system przenosi się do nowego stanu, jeżeli energia wzrasta, to nowy stan jest akceptowany zgodnie z prawdopodobieństwem danym poniższym wzorem:

$$p = e^{\frac{-\Delta E}{kT}} \tag{1}$$

gdzie: p to prawdopodobieństwo akceptacji, E – energia systemu, K – stała Boltzmana, T – temperatura.

Proces wyżarzania jest powtarzany aż do schłodzenia materiału.

Na początku lat osiemdziesiątych S. Kirkpatrick i in. [5] oraz niezależnie od nich V. Cérny [1] zaproponowali, aby algorytmu Metropolis użyć do znajdywania rozwiązań problemów optymalizacyjnych.

Ogólny algorytm symulowanego wyżarzania dla minimalizacji funkcji F, gdzie N oznacza sąsiedztwo wygląda następująco:

```
Krok 1. Stwórz rozwiązanie początkowe x0
Krok 2. Określ temperaturę początkową T0
Krok 3. Dopóki i < liczba iteracji
   Krok 3.1. licznik=0
   Krok 3.2. Dopóki licznik < nrep
      Krok 3.2.1. Wybierz rozwiązanie x z sąsiedztwa N
      Krok 3.2.2. δ = F(x) – F(x0)
      Krok 3.2.3. Jeżeli δ < 0
         x0 = x
      Krok 3.2.4. W przeciwnym wypadku
         Krok 3.2.4.1. Wylosuj liczbę b z przedziału [0,1]
         Krok 3.2.4.2. Jeżeli b < exp(-δ/T)
```

```
            x0 = x;                              //przejście „w górę”
        Krok 3.2.5. licznik = licznik + 1;
    Krok 3.3. T = A(T)                           //zmiana temperatury
```

Ważną cechą symulowanego wyżarzania jest możliwość wyboru gorszego rozwiązania. Wybór taki jest dokonywany z pewnym prawdopodobieństwem danym wzorem 1. Rolę energii pełni tutaj różnica funkcji kosztu rozwiązań δ. Dzięki temu algorytm symulowanego wyżarzania może w określonych warunkach wyjść ze znalezionego minimum lokalnego i dalej podążać w kierunku rozwiązania optymalnego.

Zmiana stanu, czyli przejście z jednego potencjalnego rozwiązania do drugiego, jest realizowana za pomocą tzw. funkcji przejścia. Proces ten polega na znalezieniu rozwiązania sąsiedniego, co jest zależne od konkretnego problemu.

Wybór temperatury początkowej oraz funkcji zmiany temperatury bezpośrednio wpływa na zachowanie się algorytmu symulowanego wyżarzania. Prawdopodobieństwo wyboru gorszego rozwiązania jest zależne od temperatury. Jeżeli końcowe rozwiązanie ma być jak najlepsze, temperatura musi być na tyle wysoka, aby umożliwić dokładne przeszukanie sąsiedztwa. Zaś zbyt wysoka temperatura początkowa może spowodować spowolnienie pracy algorytmu, poprzez przeszukiwanie zbyt dużej liczby potencjalnych rozwiązań. W początkowym etapie działania algorytmu, gdy temperatura jest wysoka, prawdopodobieństwo wyboru gorszego rozwiązania jest także wysokie. Gdy temperatura maleje, prawdopodobieństwo również. Ten etap polega na ulepszaniu istniejącego rozwiązania. To właśnie znajduje odzwierciedlenie w drugim ważnym aspekcie algorytmu symulowanego wyżarzania, czyli w powolnym ochładzaniu.

Często w algorytmie symulowanego wyżarzania stosuje się wersję z zapamiętywaniem najlepszego rozwiązania. Algorytm ten wykorzystuje jeszcze jedno rozwiązanie oprócz bieżącego i sąsiedniego. Jest to zawsze rozwiązanie najlepsze, które na początku działania algorytmu jest rozwiązaniem początkowym. W toku dalszego działania algorytmu rozwiązanie to zostaje zastąpione nowo znalezionym rozwiązaniem sąsiednim w przypadku, gdy koszt rozwiązania sąsiedniego jest niższy niż koszt rozwiązania najlepszego. Na koniec działania algorytmu zwracane jest właśnie to rozwiązanie najlepsze.

Dzięki tej modyfikacji końcowe rozwiązanie jest najlepszym rozwiązaniem znalezionym podczas działania algorytmu. Dlatego też ta wersja algorytmu została użyta w niniejszym artykule.

2. Problem dostawy z oknami czasowymi

Problem dostawy z oknami czasowymi (VRPTW – Vehicle Routing Problem with Time Windows) jest rozwinięciem problemu komiwojażera oraz jego późniejszych modyfikacji. W problemie komiwojażera (TSP – Traveling Salesman Problem) zadaniem jest znalezienie najkrótszej trasy dla zdefiniowanego zbioru klientów. Trasa komiwojażera może zaczynać się u dowolnego klienta, ale musi kończyć się w tym samym miejscu. Każdy klient może być odwiedzony jeden raz.

Rozszerzeniem problemu TSP jest problem wielu komiwojażerów (m-TSP – Multiple TSP). Różnica pomiędzy tymi problemami polega na liczbie komiwojażerów. Każdy komiwojażer rozpoczyna swoją trasę w swoim punkcie startowym i po odwiedzeniu określonej liczby klientów wraca do punktu wyjścia. Celem programu jest minimalizacja łącznej długości tras.

W problemie dostawy (VRP – Vehicle Routing Problem) zamiast komiwojażera pomiędzy klientami poruszają się pojazdy o określonej ładowności. Każdy z klientów ma określone zapotrzebowanie na towar, zaś kolejność ich odwiedzania nie ma znaczenia. Każda trasa zaczyna się oraz kończy z tego samego miejsca, jakim jest centralny magazyn. Celem jest znalezienie najkrótszej łącznej długości tras wszystkich pojazdów, przy czym ładowność żadnego z nich nie może zostać przekroczona.

Najbardziej zbliżonym do praktycznych zastosowań spotykanych w transporcie jest problem dostawy z oknami czasowymi (VRPTW – Vehicle Routing Problem with Time Windows). W problemie tym, oprócz założeń takich jak dla VRP, każdy klient ma zdefiniowany najwcześniejszy możliwy czas rozpoczęcia e_i oraz najpóźniejszy możliwy czas rozpoczęcia f_i. Wartości te wyznaczają tzw. okno czasowe klienta. Ponadto każdy klient posiada określony czas obsługi s_i, czyli okres, w jakim u klienta przebywa obsługujący go pojazd. Czas podróży pomiędzy dowolnymi dwoma klientami jest równy odległości między nimi. Jeżeli dany pojazd przyjedzie do klienta i przed czasem e_i, to musi poczekać do tego czasu z rozpoczęciem obsługi. Gdy przybędzie pomiędzy czasami e_i oraz f_i, to obsługa zaczyna się natychmiast. Jeżeli pojazd przyjedzie po czasie f_i, to klient nie może być obsłużony.

Magazyn także posiada okno czasowe, które określa maksymalny czas trwania trasy pojazdu. Znalezione rozwiązanie tego problemu powinno się charakteryzować możliwie jak najmniejszą liczbą tras oraz najmniejszą sumaryczną długością tych tras.

Wersja problemu z niedopuszczalnym spóźnieniem jest nazywana wersją z „twardymi” oknami czasowymi. W wersji z „miękkimi” oknami czasowymi dolicza się karę za spóźnienie.

Są także wersje problemu, gdzie wprowadza się rozróżnienie klientów na odbiorców i dostawców. Dla odbiorców definiuje się określoną ilość zamówionego towaru, a dla dostawców określoną ilość wolnego miejsca w pojeździe.

W niniejszym artykule rozpatrzono przypadek tylko z odbiorcami i z „twardymi" oknami czasowymi.

Problem dostawy został formalnie opisany przez Mariusa Salomona w 1987 roku. Przygotował on 56 testów, które są używane do testowania algorytmów oraz do porównywania wyników badań z aktualnie istniejącymi najlepszymi rozwiązaniami. Testy są podzielone na trzy grupy: C, R oraz RC. W testach grupy R klienci podzieleni są w sposób losowy, w grupie C klienci rozmieszczeni są w sposób uporządkowany (tzw. klastry), natomiast grupa RC jest hybrydą grup C i R, czyli część klientów jest rozmieszczona losowo, a część w sposób uporządkowany. Ponadto każda grupa jest podzielona na dwie podgrupy różniące się ładownością pojazdów.

W niniejszym artykule także użyto testów Salomona do sprawdzenia skuteczności programu.

3. Dostosowanie symulowanego wyżarzania do problemu dostawy z oknami czasowymi

Parametry algorytmu symulowanego wyżarzania można podzielić na dwie grupy. Pierwsza grupa, tzw. parametrów ogólnych polega na inicjalizacji temperatury początkowej, określeniu warunków stopu oraz planu wyżarzania. Druga grupa parametrów zależy od konkretnego problemu i są to: funkcja kosztu, funkcja przejścia i sposób tworzenia rozwiązania początkowego.

W niniejszej pracy temperatura początkowa wynosi 100 i redukowana jest wg wzoru:

$$A(T) = \alpha T \tag{2}$$

Parametr α jest doświadczalnie wybranym współczynnikiem i wynosi 0,965. Długość epoki *nrep* = 600, a liczba iteracji = 600.

Funkcja kosztu konfiguracji zwana również funkcją celu ma najczęściej postać:

$$f(s) = \alpha R + \beta D; \quad \alpha >> \beta \tag{3}$$

gdzie R oznacza liczbę tras, a D to łączna długość wszystkich tras.

W pracy przyjęto, że współczynnik β wynosi 1, a współczynnik α 10000. Dzięki tak przyjętym parametrom funkcji, minimalizowana jest przede wszystkim liczba tras.

Funkcja przejścia ma kluczowe znaczenie dla sprawności i wydajności algorytmu. Funkcja ta ma za zadanie zamieniać bieżącą konfigurację na dowolną z jej sąsiedztwa. W przy-

padku problemu dostawy, klienci na trasach są przestawiani w taki sposób, aby uzyskać pożądane zmiany konfiguracji. W niniejszej pracy wykorzystano własną funkcję przejścia.

Funkcja tworzenia rozwiązania początkowego jest punktem startowym algorytmu. Jest wiele sposobów tworzenia konfiguracji początkowej, jednakże w większości przypadków rozwiązanie początkowe ma mniejszy wpływ na wyniki końcowe niż funkcja przejścia czy też pozostałe parametry algorytmu. W tej pracy wykorzystano funkcję opracowaną przez P. Czarnasa [2] polegającą na wstawianiu klientów do tras po uprzednim ich posortowaniu według długości okna czasowego.

4. Opis zastosowanego algorytmu

4.1. Funkcja tworzenia rozwiązania początkowego

Funkcja tworzenia rozwiązania początkowego jest wykorzystywana na początku algorytmu symulowanego wyżarzania i polega na skonstruowaniu inicjalnego rozwiązania, które będzie dalej ulepszane w toku działania algorytmu. W niniejszej implementacji wykorzystano funkcję zaproponowaną przez P. Czarnasa [2], a jej ogólny algorytm jest następujący:

```
Krok 1. Posortuj klientów w kolejności rosnącej wzgl. okna czasowego
Krok 2. Dla każdego klienta
   Krok 2.1. Dla każdej trasy
      Krok 2.1.1. Znajdź najlepsze miejsce na trasie dla klienta
      Krok 2.1.2. Jeżeli nowa długość trasy < poprzednia długość
         Zapamiętaj te trasę i miejsce
   Krok 2.2. Jeżeli znaleziono miejsce wstawienia
      Wstaw klienta do tej trasy w określonym miejscu
   Krok 2.3. W przeciwnym wypadku
      Wstaw klienta na początku nowo utworzonej trasy
```

Z powodu posortowania klientów wg okna czasowego, w pierwszej kolejności do tras wstawiani są odbiorcy mający najmniejsze okna czasowe. Klientów o szerokich oknach czasowych można wstawić do kilku tras i w dodatku w wiele miejsc. Szukanie najlepszego miejsca na trasach do wstawienia posortowanych klientów jest realizowane przez funkcję opisaną w rozdziale 4.2.

4.2. Funkcja szukania najlepszego miejsca na trasie

Funkcja szukania najlepszego miejsca na trasie również została zaproponowana przez P. Czarnasa [2]. Wykorzystywana jest podczas generowania rozwiązania początkowego,

a także w funkcji przejścia. Funkcja ma na celu znalezienie najlepszego miejsca na danej trasie, do którego można wstawić danego klienta. Gdy takie miejsce istnieje, to funkcja zwraca numer pozycji dla danej trasy, gdzie należy wstawić nowego klienta.

Algorytm funkcji jest następujący:

```
Krok 1. Dla każdego klienta na trasie
   Krok 1.1. Wstaw badanego klienta za klientem bieżącym
   Krok 1.2. Jeżeli spełnione są war. czasowe i nie jest przekroczona
   ładowność pojazdu oraz
      Jeżeli nowa długość trasy < poprzednia długość
         Zapamiętaj pozycję klienta na trasie
```

Funkcja wstawia badanego klienta za każdym odbiorcą na danej trasie sprawdzając, czy nie zostały przekroczone okna czasowe oraz dopuszczalna ładowność pojazdu. Jeżeli trasa jest poprawna, to następuje porównanie długości trasy bieżącej z poprzednią.

4.3. Funkcja przejścia

Funkcja przejścia jest jednym z podstawowych elementów, który warunkuje sprawność algorytmu symulowanego wyżarzania. To od niej zależy zdolność algorytmu do wychodzenia z minimów lokalnych, co jest szczególnie istotne przy niskich wartościach temperatury. W niniejszej pracy postanowiono wykorzystać własną funkcję przejścia.

Jej schemat wygląda następująco:

```
Krok 1. Wylosuj liczbę klientów określoną parametrem TRANS_NUMBER
Krok 2. Usuń wylosowanych klientów z tras
Krok 3. Dla każdego wylosowanego klienta
   Krok 3.1. Dla każdej trasy
      Krok 3.1.1. Znajdź najlepsze miejsce dla bież. klienta na trasie
      Krok 3.1.2. Jeżeli nowa długość trasy < poprzednia długość
         Zapamiętaj tę trasę i miejsce
   Krok 3.2. Jeżeli znaleziono miejsce wstawienia
      Wstaw klienta do tej trasy w określonym miejscu
   Krok 3.3. W przeciwnym wypadku
      Wstaw klienta na początku nowo utworzonej trasy
Krok 4. Dla każdej trasy
   Krok 4.1. Ulepsz pojedynczą trasę
Krok 5. Posortuj trasy
Krok 6. Ulepsz wszystkie trasy
```

Zaletą zaprezentowanego algorytmu jest jego prostota oraz fakt losowego wyboru określonej liczby klientów. W toku badań stwierdzono, że najlepszą wartością dla parametru TRANS_NUMBER jest 10 procent liczby wszystkich klientów, czyli 10. W procesie

wstawiania wylosowanych klientów do tras bardzo duże znaczenie ma kolejność, w jakiej klienci zostali wylosowani.

Wadą algorytmu jest płytkie przeszukiwanie przestrzeni rozwiązań, co w niektórych przypadkach może skutkować gorszą jakością końcowego rozwiązania.

W kroku 4 wykonywana jest funkcja ulepszania pojedynczej trasy. Polega ona na znajdywaniu na trasie najlepszego położenia dla każdego klienta. W tym celu bieżący klient jest z tej trasy usuwany, a następnie znajdywane jest jego najlepsze położenie na tej marszrucie. Jeżeli najlepszym miejscem dla danego klienta jest jego położenie początkowe, to trasa nie zmienia się.

W kroku 5 trasy są sortowane wg liczby klientów na trasie, a w przypadku, gdy liczba klientów dla kilku tras jest taka sama, to porównywane są sumy żądań dostawy dla klientów z danej trasy. Sortowanie umożliwia ustawienie tras z najmniejszą liczbą klientów na początku, co jest bardzo przydatne w kroku 6, czyli ulepszaniu tras.

W tym etapie dla każdej trasy usuwany jest kolejno każdy klient, a następnie sprawdzana jest możliwość wstawienia tego klienta do wszystkich pozostałych tras. Ten etap umożliwia usunięcie krótkich tras, które powstały we wcześniejszych krokach. Ogólny schemat algorytmu jest następujący:

```
Krok 1. Dla każdej trasy
   Krok 1.1. Dla każdego klienta na trasie
      Krok 1.1.1. Usuń klienta z trasy
      Krok 1.1.1.1. Dla każdej trasy za wyjątkiem trasy bieżącej
         Krok 1.1.1.1.1. Znajdź najlepsze położenie klienta
         Krok 1.1.1.1.2. Jeżeli nowy koszt rozwiązania < poprzedni koszt
            Zapamiętaj te trasę i miejsce
         Krok 1.1.1.1.3. Jeżeli znaleziono miejsce wstawienia
            Wstaw klienta do tej trasy w określonym miejscu
   Krok 1.2. Jeżeli podstawowa trasa jest pusta
      Usuń trasę
   Krok 1.3. Jeżeli nie znaleziono miejsca wstawienia
      Wstaw klienta do trasy, z której go usunięto
```

5. Wyniki badań

W niniejszym rozdziale zostały przedstawione wyniki, jakie udało się uzyskać dla 56 testów M. Salomona. Wartości parametrów konfiguracyjnych były następujące: liczba iteracji algorytmu – 600, temperatura początkowa – 100, długość epoki – 600.

Uzyskane wyniki zostały porównane z najlepszymi aktualnie znanymi wynikami światowymi. Lista tych wyników jest dostępna w Internecie pod adresem:

http://www.top.sintef.no/VRP/bknown.html oraz na stronie Mariusa Salomona: http://w.cba.neu.edu/~msolomon/problems.htm.

Algorytm zaproponowany przez Piotra Czarnasa jest algorytmem zdecydowanie dokładniejszym i lepiej przeszukującym przestrzeń rozwiązań, natomiast metoda prezentowana w niniejszej pracy jest dużo łatwiejsza w implementacji oraz czas wykonywania jest stosunkowo krótki.

Tabela 1

Wyniki testów dla grupy C10x

Nazwa testu	Wyniki światowe			Uzyskane wyniki	
	Autor	Liczba tras	Dług. Tras	Liczba tras	Dług. Tras
C101	RT	10	828,94	10	828,94
C102	RT	10	828,94	10	828,94
C103	RT	10	828,06	10	828,06
C104	RT	10	824,78	10	824,78
C105	RT	10	828,94	10	828,94
C106	RT	10	828,94	10	828,94
C107	RT	10	828,94	10	828,94
C108	RT	10	828,94	10	828,94
C109	RT	10	828,94	10	828,94

Tabela 2

Wyniki testów dla grupy C20x

Nazwa testu	Wyniki światowe			Uzyskane wyniki	
	Autor	Liczba tras	Dług. Tras	Liczba tras	Dług. Tras
C201	RT	3	591,56	3	591,56
C202	RT	3	591,56	3	591,56
C203	RT	3	591,17	3	591,17
C204	RT	3	590,60	3	590,60
C205	RT	3	588,88	3	588,88
C206	RT	3	588,49	3	588,49
C207	RT	3	588,29	3	588,29
C208	RT	3	588,32	3	588,32

Tabela 3

Wyniki testów dla grupy R10x

Nazwa testu	Wyniki światowe			Uzyskane wyniki	
	Autor	Liczba tras	Dług. tras	Liczba tras	Dług. tras
R101	H	19	1645,79	19	1650,80
R102	RT	17	1486,12	17	1487,28
R103	LLH	13	1292,68	13	1304,39
R104	M	9	1007,24	9	1007,31
R105	RT	14	1377,11	14	1377,11
R106	M	12	1251,98	12	1257,96
R107	S97	10	1104,66	11	1054,98
R108	BBB	9	960,88	10	948,72

cd. tabeli 3

R109	HG	11	1194,73	12	1153,52
R110	M	10	1118,59	11	1114,43
R111	RGP	10	1096,72	10	1096,72
R112	GTA	9	982,14	9	987,24

Tabela 4

Wyniki testów dla grupy R20x

Nazwa testu	Wyniki światowe			Uzyskane wyniki	
	Autor	Liczba tras	Dług. tras	Liczba tras	Dług. tras
R201	HG	4	1252,37	4	1263,23
R202	RGP	3	1191,70	3	1191,70
R203	M	3	939,54	3	1001,78
R204	BVH	2	825,52	2	825,52
R205	RGP	3	994,42	3	1001,93
R206	SSSD	3	906,14	3	944,55
R207	BVH	2	893,33	3	943,73
R208	M	2	726,75	2	727,69
R209	H	3	909,16	3	909,16
R210	M	3	939,34	3	939,37
R211	BVH	2	892,71	3	839,03

Tabela 5

Wyniki testów dla grupy RC10x

Nazwa testu	Wyniki światowe			Uzyskane wyniki	
	Autor	Liczba tras	Dług. tras	Liczba tras	Dług. tras
RC101	TBGGP	14	1696,94	14	1703,16
RC102	TBGGP	12	1554,75	12	1554,75
RC103	S98	11	1261,67	11	1264,28
RC104	CLM	10	1135,48	10	1135,48
RC105	BBB	13	1629,44	13	1629,44
RC106	BBB	11	1424,73	11	1424,73
RC107	S97	11	1230,48	11	1230,48
RC108	TBGGP	10	1139,82	10	1139,82

Tabela 6

Wyniki testów dla grupy RC20x

Nazwa testu	Wyniki światowe			Uzyskane wyniki	
	Autor	Liczba tras	Dług. tras	Liczba tras	Dług. tras
RC201	M	4	1406,91	4	1454,51
RC202	CC	3	1367,09	3	1418,94
RC203	CC	3	1049,62	3	1093,86
RC204	M	3	798,41	3	798,46
RC205	M	4	1297,19	4	1297,65
RC206	H	3	1146,32	3	1162,16
RC207	BVH	3	1061,14	3	1105,29
RC208	IKMUY	3	828,14	3	828,71

W tabelach od 1 do 6, w pierwszej kolumnie znajdują się kolejne nazwy testów zaproponowanych przez Mariusa Salomona, kolumna „Wyniki światowe" przedstawia najlepsze rezultaty dla danego testu, natomiast w kolumnie „Uzyskane wyniki" znajdują się wyniki otrzymane za pomocą omawianego algorytmu. Tabela 7 przedstawia prace, w których te wyniki zostały zaprezentowane oraz ich autorów. Symbole w kolumnie „Autor" pochodzą od pierwszych liter nazwisk autorów danej pracy.

Tabela 7

Lista prac z najlepszymi uzyskanymi wynikami problemu dostawy

BBB	Berger J., Barkaoui M., Bräysy O.: A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows. Working paper, 2001, Defense Research Establishment Valcartier, Canada.
BVH	Bent R., Van Hentenryck P.: A Two-Stage Hybrid Local Search for the Vehicle Routing Problem with Time Windows. Technical Report CS-01-06, 2001, Department of Computer Science, Brown University.
CC	Czech Z. J., Czarnas P.A.: Parallel Simulated Annealing for the Vehicle Routing Problem with Time Windows. Proc. 10th Euromicro Workshop on Parallel, 2002, Distributed and Network-based Processing, Canary Islands, Spain, s. 376-383.
CLM	Cordeau J. F., Laporte G., Mercier A.: A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. Working Paper CRT-00-03, 2000, Centre for Research on Transportation, Montreal, Canada.
GTA	Gambardella L. M., Taillard E., Agazzi G.: MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. New Ideas in Optimization, 1999, McGraw-Hill, London, s. 63-76.
H	J. Homberger: Verteilt-parallele Metaheuristiken zur Tourenplanung. Gaber, Wiesbaden, 2000.
HG	Homberger J., Gehring H.: Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows. INFOR, 1999, VOL. 37, s297-318.
IKMUY	Ibaraki T., Kubo M., Masuda T., Uno T., Yagiura M.: Effective Local Search Algorithms for the Vehicle Routing Problem with General Time Windows. Working Paper, 2001, Department of Applied Mathematics and Physics, Kyoto University, Japan.
LLH	Li H., Lim A., Huang J.: Local Search with Annealing-like Restarts to Solve the VRPTW. Working Paper, 2001, Department of Computer Science, National University of Singapore.
M	Mester D.: An Evolutionary Strategies Algorithm for Large Scale Vehicle Routing Problem with Capacitate and Time Windows Restrictions. Working Paper, 2002, Institute of Evolution, University of Haifa, Israel.

cd. tabeli 7

RT	Rochat Y., Taillard E. D.: Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. Journal of Heuristics, 1995 1, s. 147-167.
RGP	Rousseau L. M., Gendreau M., Pesant G.: Using Constraint-Based Operators to Solve the Vehicle Routing Problem with Time Windows. Journal of Heuristics, forthcoming.
SSSD	Schrimpf G., Schneider J., Stamm-Wilbrandt H., Dueck G.: Record Breaking Optimization Results Using the Ruin and Recreate Principle. Journal of Computational Physics 159, 2000, s. 139-171.
S97	Shaw P.: A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems. Working Paper, 1997, University of Strathclyde, Glasgow, Scotland.
S98	Shaw P.: Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. Principles and Practice of Constraint Programming – CP98, 1998, Springer-Verlag, New York, s. 417-431.
TBGGP	Taillard E., Badeau P., Gendreau M., Geurtin F., Potvin J. Y.: A Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows. Transportation Science, 1997, 31, s. 170-186.

Testy z grupy C należą do najłatwiejszych i dla tej grupy program znajduje bardzo szybko rozwiązania optymalne. Testy z grup R oraz RC są testami trudniejszymi i obliczenia trwają znacznie dłużej.

Aplikacja testująca niniejszy algorytm została napisana w języku C. Podsumowując niniejsze eksperymenty, stwierdzam, że dla 28 testów osiągnięte wyniki były równe najlepszym obecnym wynikom, w 6 testach wyniki różnią się liczbą tras, pozostałe testy różnią się liczbą kilometrów.

Dużą zaletą zaproponowanego rozwiązania jest wspomniana już łatwość jego implementacji oraz stosunkowo krótki czas działania programu, dlatego wydaje się słuszny dalszy rozwój opisanej funkcji. Algorytm w większości przypadków znajduje wyniki bliskie optymalnym, ale niektóre gorsze rezultaty pokazują, że prace na funkcją przejścia powinny być ciągle kontynuowane. Obecne badania koncentrują się na rozwiązaniach hybrydowych.

LITERATURA

1. Cerny V.: A thermodynamical approach to traveling salesman problem: an efficient simulation algorithm. Journal of Optimization Theory and Applic., 1985, 45, s. 41-45.
2. Czarnas P.: Problem komiwojażerów z oknami czasowymi. Rozwiązanie metodą symulowanego wyżarzania. Uniwersytet Wrocławski, Wrocław 2001.

3. Czech Z. J., Czarnas P.: A Parallel Simulated Annealing for the Vehicle Routing Problem with Time Windows. Proc. 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing, 2002, Canary Islands, Spain, s. 376-383.
4. Kirkpatrick S., Gellat C.D., Vecchi M.P.: Optimization by simulated annealing. 1983, Science, 220, s. 671-680.
5. Metropolis N., Rosenbluth A.W., Rosenbluth M.N., Teller A.H., Teller E.: Equation of state calculation by fast computing machines. Journal of Chem. Phys., 1953, 21, s. 1087-1091.
6. Parks G.T., Suppapitnarm A., Seffen K.A., Clarkson P.J., Liu J.S.: Design by multi-objective optimization using simulated annealing. 12th International Conference in Engineering Design, 1999, 3, s. 1395-1400.
7. Reeves C.R.: Modern heuristic techniques for combinatorial problems. McGraw-Hill, 1995.
8. Solomon M.: Algorithms for the Vehicle Routing and Scheduling Problem with Time Windows Constraints. Oper. Res., 1987, 35, s. 254-265.
9. Solomon M., Desrosiers J.: Time windows constrained routing and scheduling problems. Transp. Sci., 1988, 22, s. 1-13.
10. Suppapitnarm A.: A Simulated Annealing Algorithm for Multiobjective Design Optimization. MPhil. Thesis, University of Cambridge, Cambridge 1998.
11. Ulungu E.L., Teghem J.: Multi-objective combinatorial optimization problems: a survey. Journal of multiple-criteria decision analysis, 1994, Vol. 3, s. 83-104.
12. Ulungu E.L., Teghem J.: The two phases method: an efficient procedure to solve bi-oobjective combinatorial optimization problems. Foundations of Computing and Decision Sciences, 1995, Vol. 20, No. 2, s. 149-165.
13. Ulungu E.L., Teghem J., Fortemps P.: Heuristics for multi-objective combinatorial optimization by simulated annealing. Multiple Criteria Decision Making: Theory and applications. Proceedings of the 6th International Conference on MCDM, 1995, Beijing, China, s. 228-238.

Recenzent: Dr Urszula Boryczka

Wpłynęło do Redakcji 12 lutego 2004 r.

Abstract

Article describes a new version of function generating solutions in simulated annealing algorithm for vehicle routing problem with time windows. A goal of this method is to find solutions being a good approximation of the set of efficient solutions in the short time. A method used in this paper is a modification of sequentional simulated annealing algorithm.

The presented algorithm uses concepts known from classical single criterion simulated annealing: it accepts new solutions with probability that depends on temperature and system state, it generates new solution basing on neighborhood of current solution.

Main part of new function is sampling few customers from customers list and removing them from their routes. Next the best place for each deleted customer is to be found according to procedure proposed by P. Czarnas [2]. The final part of function consists in improving all found routes by simple relocations customers between them.

Application written in C computer language was tested basing on tests proposed by M. Solomon in 1987. The majority of received solutions are comparable with world results, but few of them are much worse. All results from group C, 5 from group R and 8 from RC are as same as world results, 6 results have more routes, the rest have the same routes number but larger distance. Experiments show that above mentioned algorithm is efficient but should be still developing to receive better results.

Adres

Marcin WOCH: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16,
44-101 Gliwice, Polska, marcinwoch@wp.pl.