Problem komiwojażera dla symulowanego wyżarzania

Dominika Szydło, 250109

Czerwiec 2020

1 Problem komiwojażera (TSP)

Problem komiwojażera jest jednym z kanonicznych problemów optymalizacyjnych. Przedstawia się on następująco - komiwojażer wyrusza w trasę w celach biznesowych, podczas której planuje odwiedzić n miast. Do każdego chce zawitać jednokrotnie, za wyjątkiem miasta z którego wyrusza, ponieważ na koniec chce do niego wrócić. Celem jest wyznaczenie takiej trasy, by zminimalizować łączną przebytą odległość.

Formalnie zagadnienie to sprowadza się do znalezienia cyklu Hamiltona o najmniejszym koszcie, w grafie zupełnym ważonym o n wierzchołkach. Cykli Hamiltona w takim grafie jest $(n-1)!$, co sprawia, że wraz ze wzrostem n liczba tras do rozpatrzenia rośnie wykładniczo [8]. Należy on do grupy problemów NP-trudnych [6], a na przestrzeni lat proponowano wiele metod, bazujących na różnych heurystykach, m.in. Tabu Search [3] czy algorytmach genetycznych [1] by go rozwiązać. Aplikację w rozwiązaniu tego problemu znajduje również symulowane wyżarzanie.

2 Symulowane wyżarzanie (SA)

Optymalizacja za pomocą symulowanego wyżarzania została przedstawiona w 1983 roku przez Kirkpatricka, Gelatta i Vecchi'a [4]. Jest ona inspirowana metalurgią, w której proces wyżarzania polega na podgrzaniu metalu w celu zwiększenia jego plastyczności, by potem zastygł w strukturze krystalicznej o jak najmniejszej energii [7]. Tak samo symulowane wyżarzanie rozpoczynamy z pewną temperaturą początkową i poddajemy obróbce rozwiązanie. To czym SA się wyróżnia, to możliwość przejścia do gorszego rozwiązania - zachodzi to z pewnym prawdopodobieństwem, które maleje wraz ze spadkiem temperatury. Algorytm SA przedstawia się następująco [8]:

Algorithm 1 Symulowane wyżarzanie

Input temperatura początkowa T_0, temperatura minimalna T_{min}, maksymalna liczba iteracji K_{max}, prawdopodobieństwo spadku temperatury ρ

Output optymalne rozwiązanie x_{best}

Generowanie rozwiązania początkowego x_0;
$x_{best} \leftarrow x_0$;
Obliczenie wartości minimalizowanej funkcji $f(x_0)$ oraz $f(x_{best})$;
$T_i \leftarrow T_0$;
while $T_i > T_{min}$ **do**
  $\Delta f \leftarrow f(x_{new}) - f(x_{best})$;
  if $\Delta f < 0$ **then**
    $x_{best} \leftarrow x_{new}$;
  end if
  if $\Delta f \geqslant 0$ **then**
    $p \leftarrow e^{\frac{\Delta f}{T}}$;
    if $c \leftarrow random[0, 1] \geqslant p$ **then**
      $x_{best} \leftarrow x_{new}$;
    else
      $x_{best} \leftarrow x_{best}$;
    end if
  end if
  $i \leftarrow i + 1$;
  $T_i \leftarrow \rho \times T_i$;
end while
return x_{best}

3 Rozwiązanie problemu

3.1 Podstawowe SA

W [2] liczba miast n należy do przedziału $[10, 45]$, więc parametry SA zostały ustawione na konkretne wartości numeryczne, dające przyzwoite rezultaty dla danych tego rozmiaru. Temperatura początkowa T_0 wynosi 40000 - jest ona relatywnie niska. Również niski jest współczynnik $\rho = 0.5$, co sprawia, że temperatura dosyć szybko spada. Wprowadzono także ograniczenie liczby spadków temperatury do 50. Globalny limit iteracji K_{max} wynosi 3000, ponieważ taka liczba jest wystarczająca, by znaleźć optymalne rozwiązanie. Kolejne rozważane rozwiązania są uzyskiwane za pomocą operatorów *reverse* oraz *swap-reverse*. Nie podano niestety metody generowania rozwiązania początkowego. Algorytm testowano na losowo wygenerowanych danych o zadanym rozmiarze. Wyniki tych testów widnieją w 1.

Problem-ID	SA-Reverse Operator			SA-Swap Reverse Operator		
	Average solution	Best solution	CPU Time (secs)	Average solution	Best solution	CPU time (secs)
N10	**293.81**	**293.08**	0.127	304.59	293.08	0.186
N15	**379.62**	**374.69**	0.133	402.28	375.78	0.186
N20	**394.36**	**371.91**	0.155	485.94	419.06	0.249
N25	**457.92**	**432.22**	0.174	537.28	494.14	0.235
N30	**430.42**	**414.47**	0.189	551.34	504.58	0.251
N35	**570.25**	**531.49**	0.239	754.88	626.19	0.288
N40	**566.17**	**494.38**	0.222	791.16	713.55	0.297
N45	**601.58**	**536.58**	0.251	950.85	818.13	0.304

Rysunek 1: Wyniki testów algorytmów z [2]

3.2 SA połączone z Tabu Search

W [5] symulowane wyżarzanie zostało wzbogacone o Tabu Search w procesie generowania sąsiedztwa. Także chłodzenie zachodzi według innego równania:

$$\theta_i = \theta_{min} + \lambda \ln(1 + r_i)$$

gdzie θ_{min} jest najmniejszą wartością jaką może przyjąć temperatura, λ jest współczynnikiem kontrolującym wzrost temperatury, a r_i mówi o tym, ile iteracji minęło

od ostatniej poprawy rozwiązania. Sąsiedztwo jest generowane na dwa sposoby, w zależności od etapu poszukiwań. W fazie początkowej losowany jest fragment trasy, a następnie jest on odwracany i przesuwany w inne miejsce. Jeżeli przez X oznaczymy bieżącą trasę, a przez $Y = N(X)$ jej sąsiada, to przykładowy wynik procedury będzie miał wynik:

$$X = 1 - \underline{2 - 3 - 4 - 5} - 6 - 7 - 8 - 9 - 10$$

$$Y = 1 - 6 - 7 - 8 - \underline{5 - 4 - 3 - 2} - 9 - 10$$

W fazie końcowej, kiedy zbliżamy się do globalnego optimum, używana jest druga metoda generowania sąsiedztwa. Polega ona na zamianie kolejności dwóch miast, będących obok siebie w bieżącej trasie. Przykładowy wynik:

$$X = 1 - 2 - 3 - \underline{4 - 5} - 6 - 7 - 8 - 9 - 10$$

$$Y = 1 - 2 - 3 - \underline{5 - 4} - 6 - 7 - 8 - 9 - 10$$

Aby uniknąć badania tych samych rozwiązań, algorytm używa listy tabu, w której znajdują się ostatnio sprawdzane trasy.

Do testów algorytm ustawiono tak, by przez $\frac{3}{4}$ czasu działania generował sąsiedztwo pierwszą metodą, a przez resztę drugą. Temperatura początkowa wynosiła 1, a długość listy tabu $n \times 2.375$, gdzie n to liczba miast. Za rozwiązanie początkowe przyjęto trasę $1 - 2 - ... - n - 1$, gdzie n to liczba miast. Testy wykonano również dla klasycznego symulowanego wyżarzania ze współczynnikiem chłodzenia $\rho = 0.95$. Rezultaty przedstawia 2.

Problem	RT	opt.	meth.	best	Mean
bays29	5	2020	csa	2020	2057.8
			tssa	2028	2035.8
gr48	10	5058	csa	5125	5295.5
			tssa	5177	5235.0
eil76	10	538	csa	557	573.0
			tssa	542	564.0
berlin52	10	7542	csa	7658	7809.2
			tssa	7648	7718.5
eil51	10	426	csa	426	445.3
			tssa	430	432.5

Rysunek 2: Wyniki testów algorytmów z [5]

4 Podsumowanie

Zaprezentowane algorytmy różnią się pod dwoma istotnymi względami - zmiany temperatury oraz badania sąsiedztwa. W kwestii temperatury początkowej oraz współczynnika chłodzenia, pierwszy z nich wymaga wstępnych testów, aby można było dobrać odpowiednie parametry w zależności od instancji problemu. Drugi natomiast adaptuje się do bieżącej sytuacji, co sprawia, że jest bardziej uniwersalny. W badaniu sąsiedztwa natomiast obawą przy klasycznym SA jest generowanie zbyt dużych zmian w końcowej fazie algorytmu, a w SA z TS z kolei zbyt małych.

Oba algorytmy można wzbogacić o generowanie rozwiązania początkowego za pomocą podejścia zachłannego. Wystarczy zaimplementować prostą procedurę, która będzie generowała trasę poprzez wybieranie najbliższego miasta od tego, w którym aktualnie się komiwojażer. Nie jest ona czasochłonna i nie ma wysokiej złożoności obliczeniowej, a jest dobrym punktem wyjściowym do poszukiwań, szczególnie dla instancji, w których miasta są ułożone mniej więcej na okręgu.

Patrząc na tabelę 2 można zauważyć, że żaden z algorytmów nie przewyższa drugiego pod względem znajdowania optimum. Jeżeli popatrzymy na średnią natomiast, to dla symulowanego wyżarzania z Tabu Search jest ona niższa. Nie zmienia to jednak faktu, iż oba algorytmy są efektywne i spisują się relatywnie dobrze.

Bibliografia

[1] Ranieri Baraglia, Jose Ignacio Hidalgo i Raffaele Perego. "A hybrid heuristic for the traveling salesman problem". W: *IEEE Transactions on evolutionary computation* 5.6 (2001), s. 613–622.

[2] Mehmet DEMİRAL i Ali Isik. "Simulated Annealing Algorithm for a Medium-Sized TSP Data". W: kw. 2019. ISBN: 978-3-030-36177-8. DOI: 10.1007/978-3-030-36178-5_35.

[3] Fred Glover. "Artificial intelligence, heuristic frameworks and tabu search". W: *Managerial and Decision Economics* 11.5 (1990), s. 365–375.

[4] Scott Kirkpatrick, D. Jr i Mario Vecchi. "Optimization by Simulated Annealing". W: *Science* 220 (sty. 1983), s. 671–680. DOI: 10.1142/9789812799371_0035.

[5] Yi Liu, Shengwu Xiong i Hongbing Liu. "Hybrid simulated annealing algorithm based on adaptive cooling schedule for TSP". W: *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*. 2009, s. 895–898.

[6] Christos H Papadimitriou. “The adjacency relation on the traveling salesman polytope is NP-complete”. W: *Mathematical Programming* 14.1 (1978), s. 312–324.

[7] Zicheng Wang, Xiutang Geng i Zehui Shao. “An effective simulated annealing algorithm for solving the traveling salesman problem”. W: *Journal of Computational and Theoretical Nanoscience* 6.7 (2009), s. 1680–1686.

[8] Ai-Hua Zhou i in. “Traveling-salesman-problem algorithm based on simulated annealing and gene-expression programming”. W: *Information* 10.1 (2019), s. 7.