Dokumentacja projektu 2 NAI

Algorytm genetyczny oraz symulowanie wyżarzania dla problemu NP-trudnego – „Dyskretny Problem Plecakowy”

Autor: Bartosz Kalinowski s11027, Data: 2015-01-31

Źródła: *https://github.com/kelostrada/genetic-knapsack*

Spis treści

1. Krótki opis rozważanego problemu

Rozważany jest problem „plecakowy”. Jest to problem NP-trudny, dlatego też skorzystamy z rozwiązań genetycznych. Wariant problemu jaki autor próbuje zbadać to najprostszy z możliwych wariant z jednym plecakiem który posiada określoną wytrzymałość (maksymalną wagę przedmiotów które może przechować). Zadany jest także zbiór przedmiotów, każdy z określoną wartością i wagą które próbujemy schować do plecaka. Przedmioty oczywiście raczej nie powinny dać się schować wszystkie na raz do plecaka i problemem jest wybranie takiego układu przedmiotów, żeby schować do plecaka przedmioty o jak największej wartości, przy tym nie przekraczając zadanego limitu wagowego.

Formalnie problem może być zdefiniowany:

Mamy do dyspozycji plecak o maksymalnej pojemności B oraz zbiór N elementów $\{x_1, x_j, ..., x_N\}$, przy czym każdy element ma określoną wartość c_j oraz wielkość w_j.

Zmaksymalizuj: $\sum_{j=1}^{N} c_j x_j .$

przy założeniach: $\sum_{j=1}^{N} w_j x_j \leq B, \qquad x_j = 0 \text{ lub } 1, \quad j = 1, \ldots, n.$

2. Opis rozwiązań technicznych

2.1 Reprezentacja zbioru przedmiotów

Zbiór przedmiotów chowanych do plecaka przechowywany jest w formie tablicy ze strukturami posiadającymi dwa pola – wartość i waga. Kiedy któreś z rozwiązań potrzebuje sprawdzić czy jest poprawne i czy przedmioty mieszczą się do plecaka musi odwołać się do tej tablicy.

2.2 Reprezentacja rozwiązań (Kodowanie rozwiązań w strukturze chromosomu)

Autor reprezentuje rozwiązania przy pomocy wektora binarnego. Każda jedynka na wektorze odpowiada informacji, że dany przedmiot został wybrany do bycia włożonym do plecaka.

Przykładowy wektor: **(1,0,0,1,1)**

Taki wektor oznacza, że wybraliśmy przedmioty z indeksów 0, 3, 4 i zostały one włożone do plecaka.

2.3 Relacja sąsiedztwa

W przypadku algorytmów optymalizacyjnych takich jak właśnie rozważany algorytm symulowanego wyżarzania potrzebujemy zdefiniować relacje sąsiedztwa pomiędzy układami rozwiązań. W rozważanym problemie zastosowana została relacja sąsiedztwa która bazuje na mechanizmie mutacji z implementacji klasycznego algorytmu genetycznego (opis dalej). Najprościej jednak ujmując dany wektor jest w sąsiedztwie ze wszystkim wektorami które można utworzyć poprzez zastosowanie na nim jednej mutacji.

2.4 Dane testowe

Zbiór danych testowych do został wygenerowany losowo. Ten sam zbiór populacji wykorzystany zostanie przy algorytmie genetycznym co przy algorytmie symulowanego wyżarzania.

3. Opis rozwiązań zastosowanych w algorytmach

3.1 Opis algorytmu genetycznego

Użyty algorytm składa się z kilku prostych kroków.

1. Generujemy początkową losową populację rozwiązań
2. Wybieramy dwa najlepsze rozwiązania z populacji i zapisujemy do nowego zbioru populacji
3. Losujemy dwukrotną ilość par rozwiązań ile jest elementów w podstawowej populacji
4. Stosujemy krzyżówkę między tymi parami i generujemy dwójkę dzieci
5. Jeżeli zostanie spełniony warunek losowy (domyślnie 5%), to zostanie zastosowana mutacja na dzieciach wynikających z krzyżówki
6. Jeżeli dzieci spełniają warunki (są rozwiązaniami) i nie ma ich w zbiorze nowej populacji to dokładamy je do nowej populacji
7. Sortujemy nową populację względem najlepszych rozwiązań i wyciągamy tyle ile było w podstawowej populacji
8. Nadpisujemy starą populację nową i jeśli najlepsze rozwiązanie nie powtórzyło się n razy (domyślnie 100) to wracamy do kroku 2.

3.2 Opis algorytmu wyżarzania

Algorytm wyżarzania opiera się na idei, że wraz z upływającym czasem temperatura układu coraz bardziej wygasa i dzięki temu im jest późniejszy etap algorytmu tym mniejsza szansa żeby wejść na nową ścieżkę. Jeśli chodzi o ścieżkę to chodzi o ścieżkę w kontekście algorytmu wspinaczkowego.

3.3 Operacja krzyżowania – Crossover

Zastosowana została standardowa operacja krzyżowania dla klasycznego algorytmu genetycznego. Tzn podczas krzyżowania następuje losowanie indeksu i złączenie dwóch wektorów na tym indeksie. Tzn dostajemy dwa nowe wektory będące dziećmi poprzednich, mające jedną część jednego i drugą drugiego wektora.

3.4 Operacja mutacji – Mutation

Zastosowana operacja mutacji jest bardzo prosta. Losujemy dwa indeksy w wektorze rozwiązań i zamieniamy na nich wartości. Nic więcej się nie dzieje, także czasami mutacja może nawet nie wprowadzić żadnych zmian. Taka interpretacja mutacji oznacza po prostu wyciągnięcie jednego przedmiotu z plecaka i włożenie innego.

3.5 Metoda selekcji

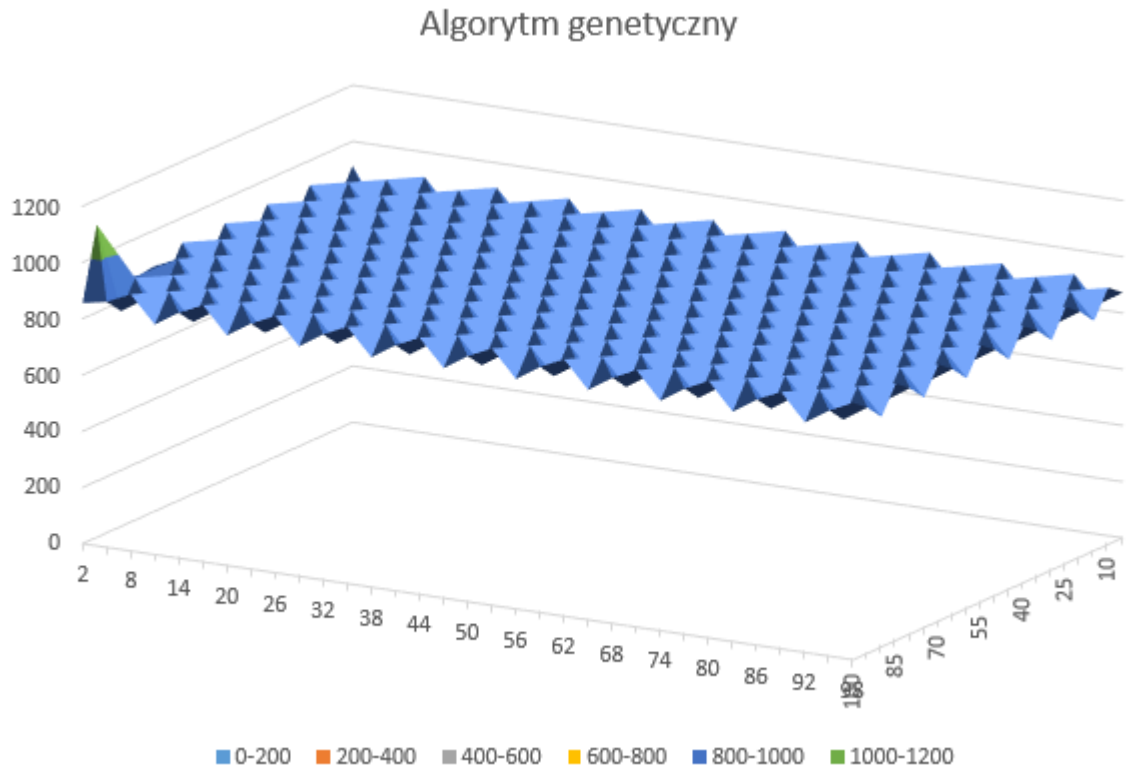
Rozwiązania porównywane są w bardzo prosty sposób. Sumowana jest wartość każdego przedmiotu który został włożony do plecaka. Jeżeli maksymalny rozmiar plecaka zostanie przekroczony (jeśli chodzi o wagi) to wartość podawana jest 0. Jeżeli nie przekroczymy rozmiaru plecaka to suma wartości jest wynikiem tej funkcji. Potem na podstawie tego wyniku porównujemy różne rozwiązania w celu sprawdzenia które rozwiązanie jest „lepsze”.

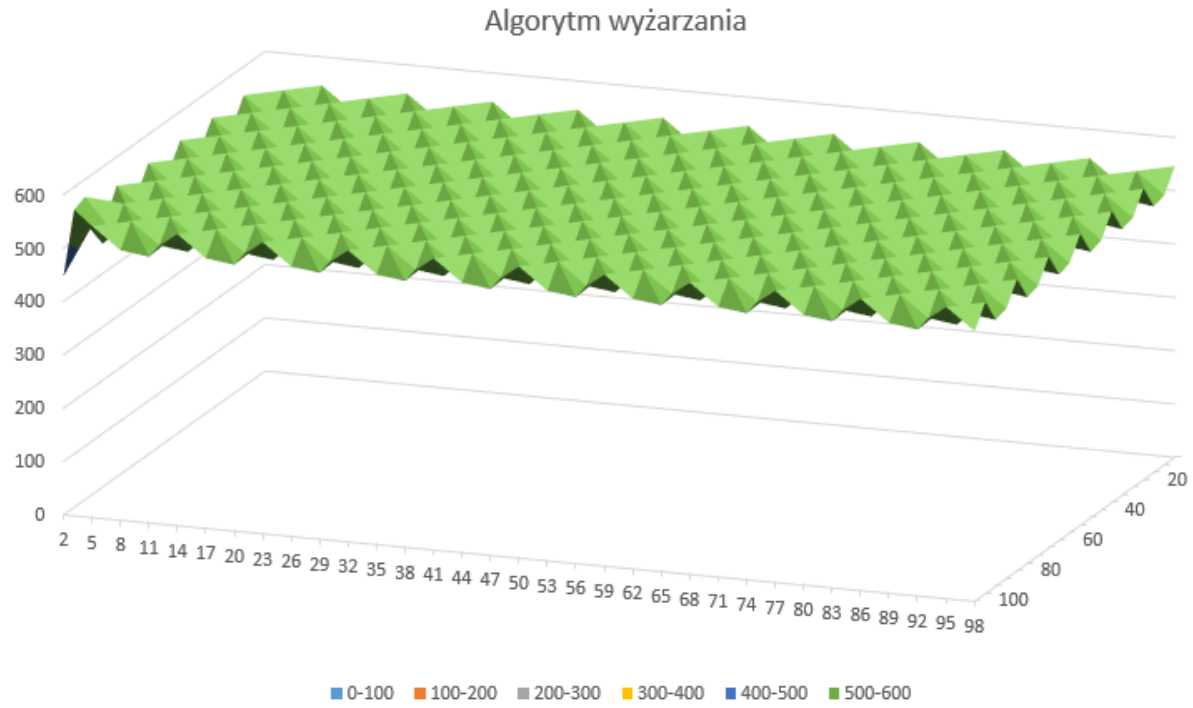
3.6 Funkcja wyżarzania

Zastosowana funkcja wyżarzania opiera się na prostej funkcji $f(k) = \frac{T_0}{k}$, gdzie T_0 to temperatura początkowa.

4. Wykresy i wnioski

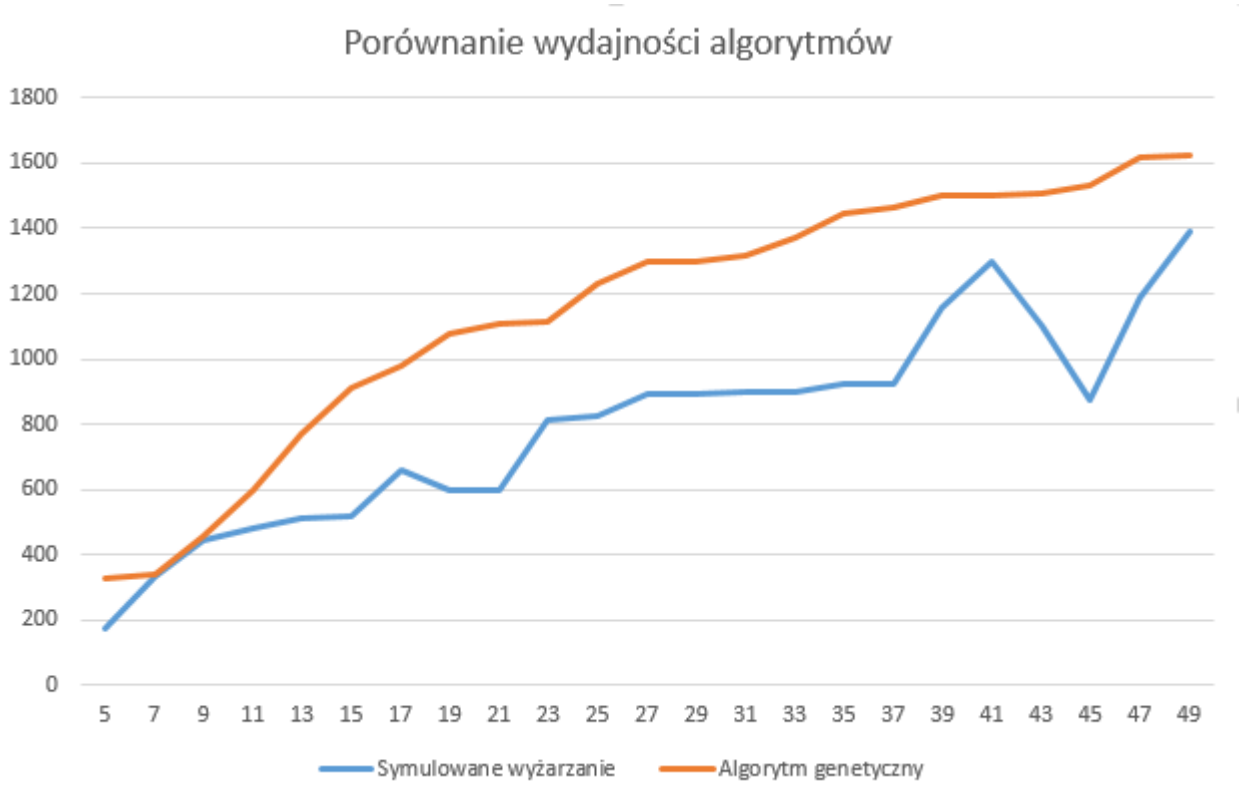
4.1 Zależność wydajności algorytmów od szansy na mutacje i wielkości populacji





Jak widać, zmiana szansy na mutacje i rozmiar populacji nie miały w zasadzie żadnego wpływu na wydajność algorytmu. Uzyskane wyniki nie różnią się od siebie, a wręcz widać pewien schemat w jaki się one układają i tworzą pewnego rodzaju płaszczyznę. Tylko w przypadku algorytmu genetycznego udało się uzyskać lepsze wyniki przy bardzo wysokiej szansie na mutacje i niewielkiej populacji. Jednak mogło być to spowodowane zwykłym przypadkiem.

4.2 Wykresy wydajności algorytmu w zależności od wielkości badanego problemu



Jak widać z wykresu, algorytm genetyczny zachował się dużo wydajniej od symulowanego wyżarzania. W podobnym czasie osiągał dużo lepsze wyniki od wyżarzania. Nie oznacza to oczywiście, że wyżarzanie było kompletnie bezużyteczne – nadal potrafiło wskazać wyniki które być może w pewnych warunkach byłyby zadowalające. Jednak zaprezentowana implementacja algorytmu genetycznego sprawdziła się dużo bardziej.

Na wykresie na dolnej belce widać liczby badanych przedmiotów które mogą być przechowywane w plecaku.

4.3 Wnioski

Jak zauważono w poprzednim punkcie, algorytm genetyczny zachowuje się dużo wydajniej dla badanego problemu. Okazało się, że manewrowanie parametrami sterującymi nie miało większego wpływu na wydajność algorytmu. Oczywiście przy większych populacjach długość wykonywania się algorytmu drastycznie wzrastała, ale nie przeniosło się to wcale na dużo lepsze wyniki niż w przypadku mniejszych populacji.

W związku z powyższym ciężko ustalić jest jednoznacznie jakieś optymalne parametry do sterowania algorytmami.