POLITECHNIKA BIAŁOSTOCKA

WYDZIAŁ INFORMATYKI

PRACA DYPLOMOWA INŻYNIERSKA

TEMAT: TEMAT PRACY INŻYNIERSKIEJ / MAGISTERSKIEJ.

WYKONAWCA: GAL ANONIM

.....

podpis

PROMOTOR: DR INŻ. DOKTOR INŻYNIER

.....

podpis

BIAŁYSTOK 2021 r.

Załącznik nr 2 do „Zasad postępowania przy przygotowaniu i obronie pracy dyplomowej na PB”

Karta dyplomowa

POLITECHNIKA BIAŁOSTOCKA Wydział.....	Studia..... stacjonarne/niestacjonarne ..... studia I stopnia/studia II stopnia	Nr albumu studenta..... Rok akademicki..... Kierunek studiów..... ..... Specjalność..... .....

.....

Imiona i nazwisko studenta

TEMAT PRACY DYPLOMOWEJ:

.....

.....

.....

Zakres pracy:

1. .....
2. .....
3. .....
4. .....

Słowa kluczowe (max 5): .....

TO JEST SKAN

.....

Imiona i nazwisko, stopień/ tytuł promotora - podpis

.....	.....	.....
Data wydania tematu pracy dyplomowej - podpis promotora	*Regulaminowy termin złożenia pracy dyplomowej*	*Data złożenia pracy dyplomowej - potwierdzenie dziekanatu*

.....	.....
Ocena promotora	*Podpis promotora*

.....	.....	.....
Imiona i nazwisko, stopień/ tytuł recenzenta	*Ocena recenzenta*	*Podpis recenzenta*

Subject of diploma thesis

Temat po angielsku.

Summary

Streszczenie pracy po angielsku.

Załącznik nr 4 do „Zasad postępowania przy przygotowaniu i obronie pracy dyplomowej na PB”
Białystok, dnia 05.01.2020 r.

Gal Anonim
Imiona i nazwisko studenta
12345
Nr albumu
informatyka, stacjonarne
Kierunek i forma studiów
dr inż. Doktor Inżynier
Promotor pracy dyplomowej

OŚWIADCZENIE

Przedkładając w roku akademickim 2019/2020 Promotorowi **dr inż. Doktor Inżynier** pracę dyplomową pt.: **Temat pracy**, dalej zwaną pracą dyplomową, **oświadczam, że**:

1) praca dyplomowa stanowi wynik samodzielnej pracy twórczej;
2) wykorzystując w pracy dyplomowej materiały źródłowe, w tym w szczególności: monografie, artykuły naukowe, zestawienia zawierające wyniki badań (opublikowane, jak i nieopublikowane), materiały ze stron internetowych, w przypisach wskazywałem/am ich autora, tytuł, miejsce i rok publikacji oraz stronę, z której pochodzą powoływane fragmenty, ponadto w pracy dyplomowej zamieściłem/am bibliografię;
3) praca dyplomowa nie zawiera żadnych danych, informacji i materiałów, których publikacja nie jest prawnie dozwolona;
4) praca dyplomowa dotychczas nie stanowiła podstawy nadania tytułu zawodowego, stopnia naukowego, tytułu naukowego oraz uzyskania innych kwalifikacji;
5) treść pracy dyplomowej przekazanej do dziekanatu Wydziału Informatyki jest jednakowa w wersji drukowanej oraz w formie elektronicznej;
6) jestem świadomy/a, że naruszenie praw autorskich podlega odpowiedzialności na podstawie przepisów ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. z 2019 r. poz. 1231, późn. zm.), jednocześnie na podstawie przepisów ustawy z dnia 20 lipca 2018 roku Prawo o szkolnictwie wyższym i nauce (Dz. U. poz. 1668, z późn. zm.) stanowi przesłankę wszczęcia postępowania dyscyplinarnego oraz stwierdzenia nieważności postępowania w sprawie nadania tytułu zawodowego;
7) udzielam Politechnice Białostockiej nieodpłatnej, nieograniczonej terytorialnie i czasowo licencji wyłącznej na umieszczenie i przechowywanie elektronicznej wersji pracy dyplomowej w zbiorach systemu Archiwum Prac Dyplomowych Politechniki Białostockiej oraz jej zwielokrotniania i udostępniania w formie elektronicznej w zakresie koniecznym do weryfikacji autorstwa tej pracy i ochrony przed przywłaszczeniem jej autorstwa.

.....

czytelny podpis studenta

Spis treści

Wstęp

wstep, wejscie do problemu, zalew smieci, cel pracy i zakres, jak jest zorganizowana

1. Ogólny problem

Badany przez nas problem jest w informatyce nazywany problemem komiwojażera. W tym rozdziale zostanie on omówiony oraz metody optymalizacji.

1.1 Problem komiwojażera

Problem komiwojażera (ang. travelling salesman problem - TSP) należy do rodziny problemów NP-trudnych. Znalezienie najlepszego rozwiązania dla tego problemu jest trudne i fascynuje naukowców od wielu lat. Niektórzy poddają pod wątpliwość znalezienie efektywnego rozwiązania czyli takiego którego czas działania jest maksymalnie wielomianowy. Aktualnie istnieje wiele rozwiązań tego problemu, a proponowane podejścia są bardzo interesujące. Niektóre z nich bazują na lokalnych przeszukiwaniach grafu, a inne opierają się na rozwoju.

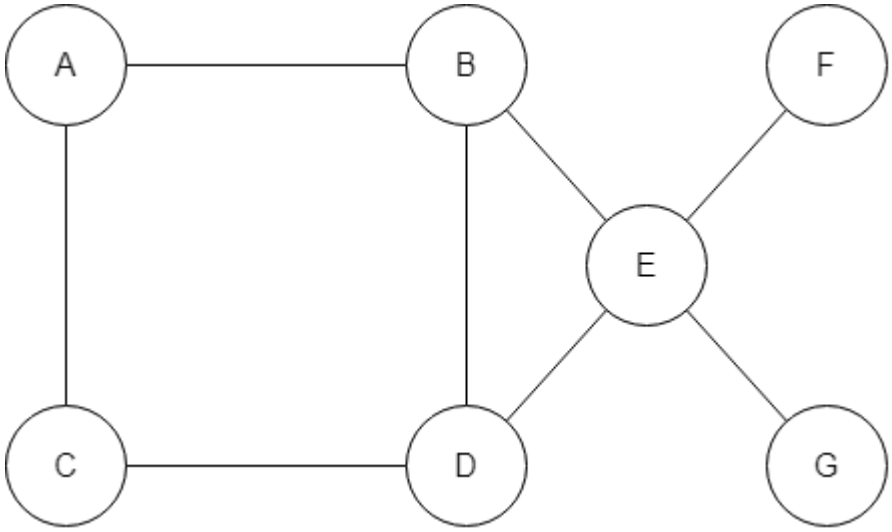
Podobnym problemem do TSP jest problem konika szachowego. Problem ten jest problemem NP-zupełnym. Już w XVIII wieku badania nad tym problemem rozpoczął Euler. Rozwiązanie tego problemu polega na znalezieniu ścieżki jaką ma przebyć konik szachowy, tak aby odwiedzić każde pole na szachownicy tylko i wyłącznie raz. Skoczek porusza się po planszy zgodnie z określonym ruchem, a plansza szachowa może mieć różny rozmiar. Konik porusza się aż do momentu odwiedzenia wszystkich pól lub do momentu w którym nie ma możliwości odwiedzenia kolejnego pola.

Optymalizacja tras od zawsze jest obecna w historii ludzkości. Nawet takie trywialne problemy jak podróż pomiędzy 3 miejscowościami może zostać sklasyfikowany jako problem komiwojażera. Chociaż dokładne wskazanie na źródło problemu TSP nie jest znane, to już w 1832 roku w przewodniku dla podróżujących po Niemczech i Szwajcarii została zawarta informacja o optymalizacji trasy przejazdu. Nie ma tam zawartych żadnych teorii matematycznych w związku z czym nie można uznać tego dzieła za początek rozważań nad problemem komiwojażera.

W XIX wieku William Hamilton stworzył fundamenty pod definicję TSP. W rozwiązaniu problemu komiwojażera należy znaleźć cykl w grafie. W skład takiego cyklu musi zostać zawarty każdy z wierzchołków. Każdy z wierzchołków może znajdować się w rozwiązaniu dokładnie tylko raz. Cykl który spełnia wymieniony warunek jest cyklem

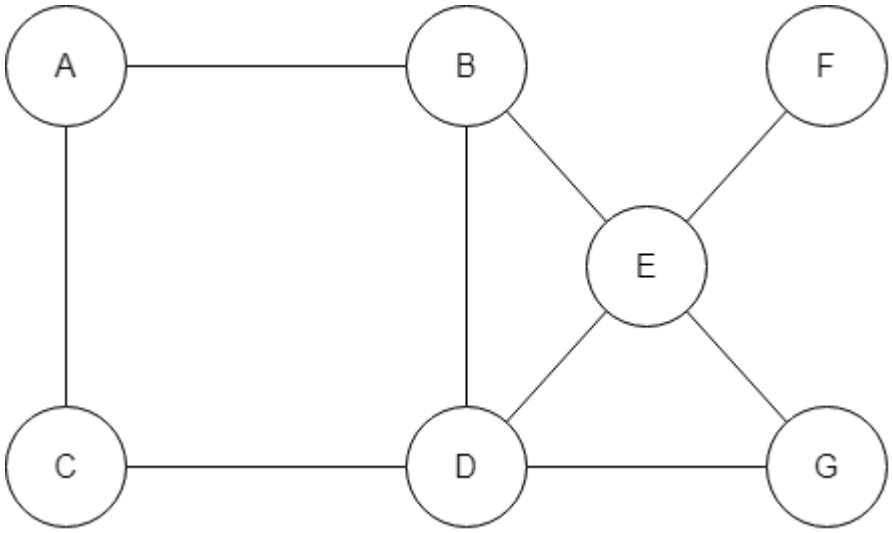
Hamiltona. Jeśli w grafie można wyróżnić cykl z opisanymi powyżej warunkami, to graf jest grafem Hamiltonowskim.

Na 1.1 został przedstawiony graf bez cyklu Hamiltona. W grafie tym nie można znaleźć takiego połączenia które zawiera wszystkie wierzchołki przechodząc przez każdy z nich dokładnie raz. Istnieje możliwość przejścia przez wszystkie wierzchołki jedynie po powtórnym odwiedzeniu przynajmniej jednego wierzchołka.



Rysunek 1.1: Graf bez cyklu Hamiltona.

Graf z 1.2 posiada połączenie krawędzi dzięki któremu można przejść po wszystkich wierzchołkach dokładnie raz. Takie przejście jest właśnie cyklem Hamiltona w związku z czym graf jest Hamiltonowski. Wyruszając przykładowo z punktu F możemy przejść kolejno do E - G - D - B - A - C. W ten sposób odwiedzimy wszystkie wierzchołki tylko raz.



Rysunek 1.2: Graf z cyklem Hamiltona.

W latach 30 XX wieku Merrill Meeks Flood rozpoczął rozważania nad optymalizacją przejazdu autobusów szkolnych. Działalność tą możemy uznać za początek pracy nad problemem TSP. Wraz z upływem czasu zainteresowanie problemem optymalizacyjnym narastało a co za tym idzie powstawały nowe pomysły na algorytmy. Jednak żaden z pomysłów nie jest w stanie zaproponować dokładnego rozwiązania które jest w stanie przedstawić rezultat w czasie wyrażonym za pomocą wielomianu.

1.2 Metody optymalizacji

Jednym z proponowanych rozwiązań jest algorytm Helda Karpa który jest oparty na programowaniu dynamicznym. Złożoność pamięciowa tego algorytmy wynosi O(n razy 2 do n), a czasowa O(n do 2 razy 2 do n). W algorytmie tym na każdym kroku wyznaczamy punkt który powinien być przedostatni na trasie. Aby wyznaczyć poprzednika należy skorzystać ze wzoru w którym poszukiwana jest najmniejsza wartość pomiędzy punktami.

Innym przykładem , który można wykorzystać do rozwiązania problemu komiwojażera jest algorytm najbliższego sąsiada. Rozwiązanie to wykorzystuje strategię zachłanną. W algorytmie szukamy aktualnie najlepszego ruchu. W tym celu przeszukiwani są jedynie sąsiedzi którzy są najbliżej aktualnego punktu. Złożoność takiego algorytmu jest szacowana na O(n do 2).

Oprócz standardowych przeszukiwań zbiorów na przestrzeni lat pojawiły się propozycje które wprowadzają elementy losowości. Przykładem takiego rozwiązania mogę być algorytm genetyczny oraz algorytm mrówkowy. Należą one do grup algorytmów heurystycznych, czyli do takich, które nie dają gwarancji znalezienia dokładnego rozwiązania.

W pracy zostaną zbadane rozwiązania problemu za pomocą algorytmu genetycznego, mrówkowego oraz zachłannego. W pozostałych podrozdziałach zostaną opisane ich klasyczne wersje.

1.2.1 Algorytm genetyczny - Paweł

Algorytm genetyczny(ang. Genetic Algorithm - GA) polega na symulacji procesów genetycznych. W przyrodzie często najsłabsze osobniki, nie biorą udziału w reprodukcji, przez co nie mogą przekazać swoich słabych genów potomkom. Podobnie w algorytmie genetycznym populacja poddawana jest operatorom genetycznym: selekcja najlepszych osobników, krzyżowanie oraz mutacja. Algorytm dąży do tego, aby początkowe pokolenia

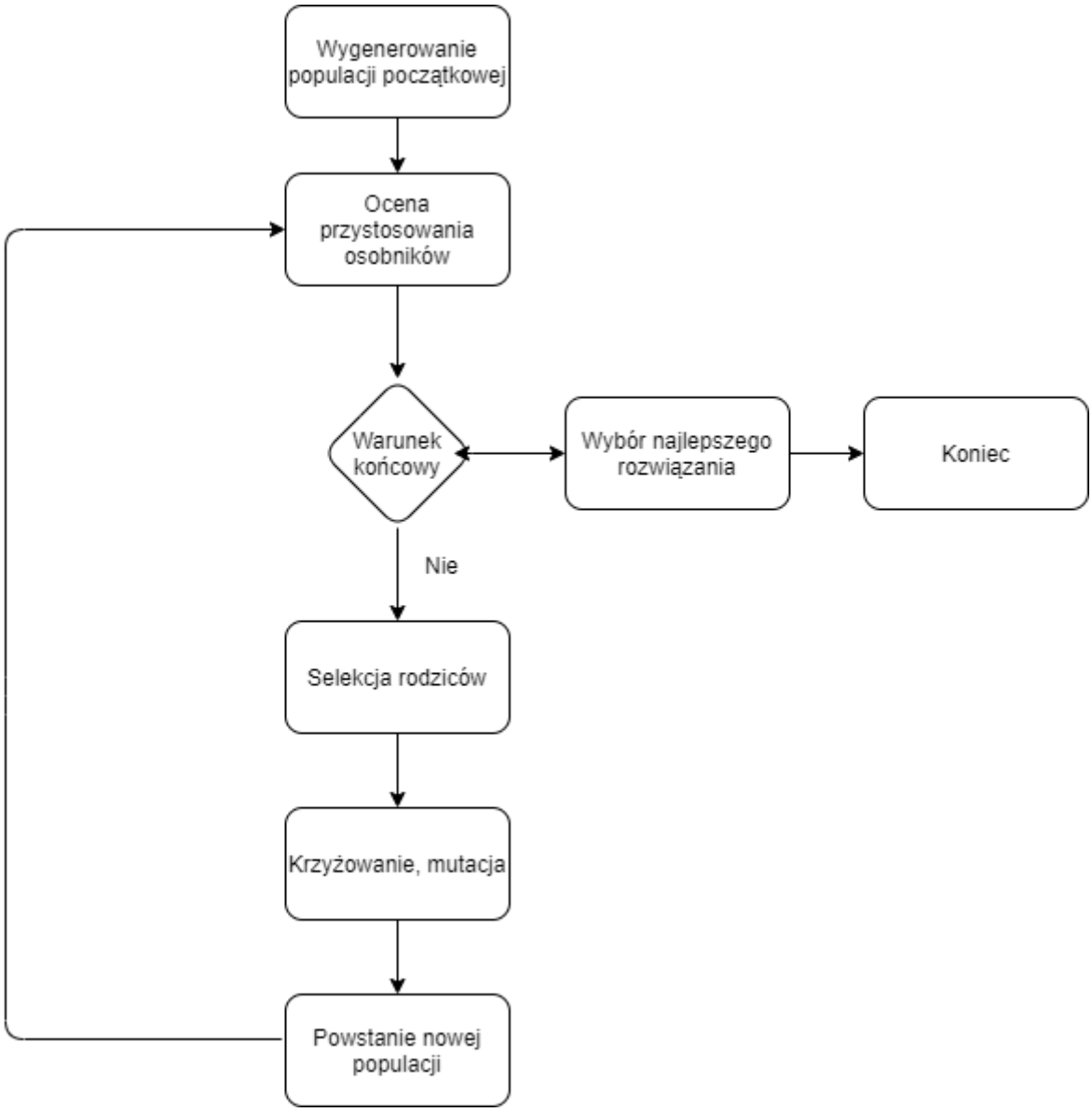
z kolejnymi iteracjami ewoluowały w coraz to lepsze rozwiązania. Algorytm zawiera wiele elementów losowych np. rozmiar populacji, ilość pokoleń lub prawdopodobieństwo krzyżowani/mutacji. Te czynniki powodują, że wyniki za każdym razem mogą zdarzać się inne. Wszystko zależy od złożoności badanego problemu oraz danych.[**?**].

Algorytmy genetyczne są od dawna stosowane informatyce do rozwiązywania problemów komiwojażera oraz innych NP trudnych zagadnień. Pionierem algorytmów genetycznych był John Henry Holland[**?**], który w latach 70 napisał książkę o algorytmach ewolucyjnych Ądaptation in Natural and Artificial Systems". Medycyna jest jedną z ważniejszych dziedzin, gdzie wykorzystuje się algorytmy genetyczne. Zbiory danych i przestrzeń przeszukiwań jest ogromna i złożona. Zazwyczaj w oparciu o te informację, lekarz musi podjąć decyzję, czy np. nowotwór jest złośliwy, czy ma łagodny przebieg. Algorytm genetyczny pozwala wspomóc lekarza przy podejmowania takich decyzji, przetwarzając i analizując te ogromne zbiory. [**?**]. Kolejną gałęzią gospodarki, w których zastosowanie znajduje algorytm genetyczny, jest przemysł spożywczy, a konkretnie optymalizacja linii produkcyjnych. Za ich pomocą algorytmu genetycznego wyznacza się parametry takie jak temperatura, ciśnieniu lub zapotrzebowanie mocy. Dzięki temu wszystkie procesy technologiczne zachodzące w maszynach i urządzeniach są wydajne i zoptymalizowane[**?**]. Algorytmy genetyczne często stosuje się jako wskazanie punktów początkowych w innych metodach optymalizacyjnych. Poza podanymi przykładami, znajdują one zastosowanie praktycznie wszędzie: ekonomia, giełda, przemysł lotniczy, kombinatoryka, sieci komputerowe, zarządzanie łańcuchem dostaw a także ustalanie czasu reklamy w telewizji [**?**].

Przed przejściem do omawiania algorytmu, należy wyjaśnić podstawowe pojęcia, które występują w algorytmie genetycznym:

- OSOBNIK pojedyncze rozwiązanie problemu, zakodowane w postaci chromosomu.
- POPULACJA zbiór osobników o stałej liczbie N w przekroju trwania całego algorytmu.
- GEN przechowuje informację o dowolnej cesze osobnika. W zależności od sposobu kodowania może to być bit, dowolna cyfra, znak itp.
- CHROMOSOM składa się z uporządkowanego ciągu genów, przechowuje wszystkie cechy osobnika
- GENOTYP w przyrodzie może składać się z kilku chromosomów i określa skład osobnika. W algorytmach genetycznych przyjmuje się, że jest to pojedynczy chromosom [**?**].

• FUNKCJA PRZYSTOSOWANIA



Rysunek 1.3: Schemat algorytmu genetycznego

Schemat blokowy klasycznego algorytm genetycznego został pokazany na rysunku 1.3

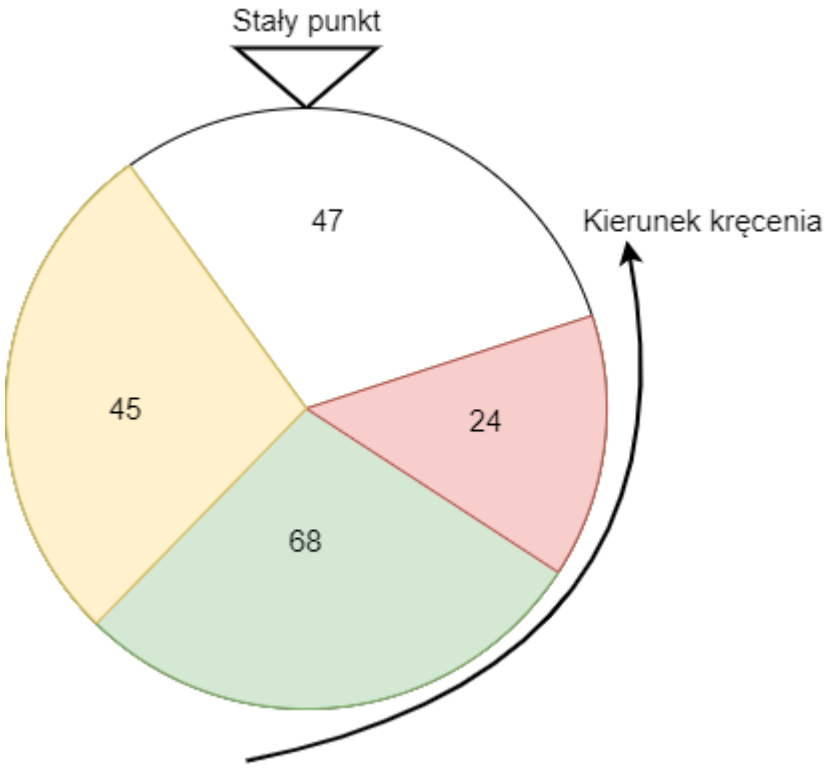
Pierwszym krokiem jest wylosowanie populacji początkowej algorytmu. Wielkość populacji podczas trwania całego algorytmu jest stała N. Ważne jest, aby wszystkie osobniki były jak najbardziej zróżnicowane i wygenerowane losowo. Każdy z nich następnie musi zostać zakodowany do postaci chromosomów, które będą przechowywać w sobie informację o odwiedzanych punktach w postaci genów.

W populacji każdy osobnik musi zostać poddany ocenie funkcji przystosowania. Jej wynik determinuje jak dobre jest dane rozwiązanie. W klasycznym algorytmie dąży

się do maksymalizacji tej funkcji. Określenie jak dana funkcja przystosowania będzie wyglądać, jest to jedną z najważniejszych części algorytmu genetycznego. Jeśli zostanie źle zdefiniowana, znalezione osobnik może nie spełniać wymagań rozwiązania problemu.

Po ocenie osobników zostaje sprawdzony warunek końcowy algorytmu. W zależności od problemu zostaje zdefiniowany inny warunek. W klasycznych podejściach są dwa rodzaje warunków końcowych. Pierwszym popularnym warunkiem końcowym jest stała ilość iteracji algorytmu, czyli po wykonaniu określonej ilości razy ewolucji, wybierany jest najlepszy osobnik z populacji. Drugim warunkiem zazwyczaj jest przetwarzanie algorytmu dopóki nie zostanie znaleziony dostatecznie dobry osobnik. Należy również założyć, że jeśli w kolejnych pokoleniach nie zachodzi poprawa najlepszego rozwiązania, należy przerwać. Wybór w jaki sposób będzie wyglądać warunek końcowy zależy od wielu czynników. Jeśli ważny jest krótki czas, należy założyć pierwszy wariant. Jeśli natomiast algorytm może szukać rozwiązania nawet kilka godzin, można przyjąć drugi wariant.

Kolejnym krokiem algorytmu jest wyselekcjonowanie rodziców do reprodukcji. Polega ona na tym, że osobniki lepsze(mają większą wartość oceny przystosowania)mają większe szansę na pozostanie rodzicem i przekazanie swoich cech. [**?**]. Najpopularniejszymi metodami wyboru rodziców jest metoda ruletki oraz turniejowa.

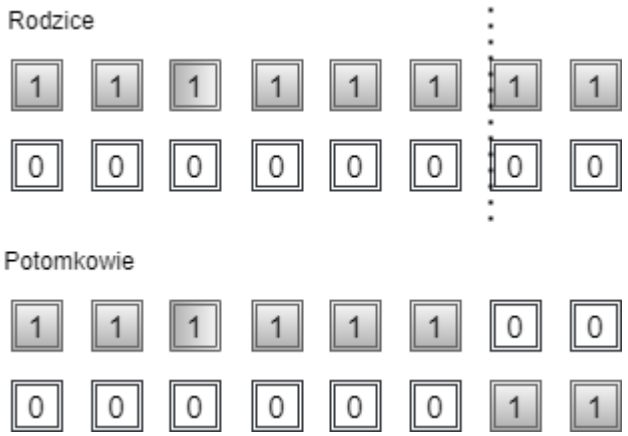


Rysunek 1.4: Metoda ruletki

Na rysunku 1.4 została zilustrowana pierwsza metoda ruletki. Każdy z osobników dostaje wirtualny wycinek koła fortuny. Jego wielkość zależy od wartości funkcji prawdopodobieństwa. Przy każdym wyborze rodzica następuje zakręceniem koła i do reprodukcji zostaje wybrany osobnik na który będzie wskazywał stały punkt.

W metodzie turniejowej zostaje wybranych r osobników z populacji N. Z pośród nich zostaje wybrany zwycięzca(największa wartość funkcji przystosowania), który trafia do puli rodzicielskiej. Im większa jest ilość osobników r tym mniejsze szanse, że słabsze osobniki zostaną wybrane.

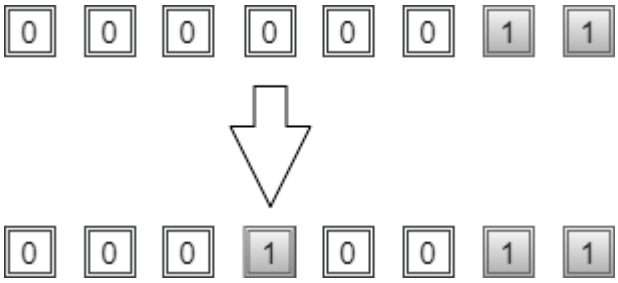
Wybrani rodzice zostają poddani operatorom genetycznym: krzyżowanie(ang. crossover) oraz mutacji(ang. mutation). Krzyżowanie polega na połączeniu części chromosomu jednego rodzica z częścią drugiego. Wynikiem takiego połączenia jest nowy osobnik. Proces krzyżowania w zależności może przebiegać w różny ale zawsze określony sposób. Wszystko zależy od metody zakodowania chromosomu oraz od tego czy kolejność genów i ich unikalność ma znaczenie. W klasycznym podejściu polega na rozcięciu w dowolnym miejscu genotypu u dwóch osobników oraz skrzyżowaniu ich ze sobą w tym punkcie(rys. 1.5). Następnie u nowego osobnika może z prawdopodobieństwem pm wystąpić mutacja.



Rysunek 1.5: Klasyczne krzyżowanie

Jest to zmiana dowolnego pojedynczego rys 1.6 lub ciągu genów na inny. Wartość pm w klasycznych algorytmach jest stosunkowo niskie. Mutacja ma na celu delikatne zróżnicowanie osobników w celu przeszukania nowej przestrzeni rozwiązań. Natomiast gdyby zachodziła często, mogłaby powodować niszczenie dobrych rozwiązań.

Po zastosowaniu wszystkich operatorów genetycznych, nowa populacja jest poddawana ocenie przystosowania i jeśli wystąpił warunek końcowy, wybierane jest najlepsze



Rysunek 1.6: Mutacja genotypu

rozwiązanie. Algorytmy genetyczne i jego odmiany zrewolucjonizowały systemy informatyczne. Nie są one algorytmami, które wyliczają dokładne wyniki, ale przy odpowiedniej implementacji i ustaleniu parametrów wyjściowych, pozwalają osiągnąć wystarczające rezultaty. Bardzo ważny jest czas znalezienia takiego rozwiązania. Jeśli rozwiązanie idealne obliczane jest w ciągu 24 godzin przez algorytm dokładny, a algorytm genetyczny w trakcie kilku minut znajdzie rozwiązanie będące w sąsiedztwie, swoją efektywnością wygra ten drugi, gdyż użytkownik, nie będzie chciał czekać całej doby na wynik swojego zapytania. Oczywiście, algorytmy genetyczne też mogą znaleźć najlepsze rozwiązanie. Kwestią ograniczenia jest zawsze czas.

1.2.2 Algorytm mrówkowy

Obserwacje nad zachowaniami w przyrodzie wielokrotnie miały wpływ na rozwój nowych rozwiązań. Tak jak w przypadku algorytmu genetycznego, tak i w przypadku algorytmu mrówkowego pomysł został zaczerpnięty z przyrody. Dokładne działanie algorytmu mrówkowego wzoruje się na zachowaniu kolonii mrówek. Dzięki pracy zespołowej, owady te są w stanie wypracować optymalną ścieżkę między siedliskiem a znalezionym pokarmem.

Dla niejednego gatunku problematyczne mogłoby być odtworzenie przebytej ścieżki. Na początku należałoby zadać sobie pytanie w jaki sposób te niewielkich rozmiarów owady są w stanie znacząco ułatwić sobie przetrwanie? Istotną rolę odgrywa tutaj wspomniana już praca zespołowa. To dzięki współpracy mrówki są w stanie optymalizować trasę. Innym ważnym czynnikiem determinującym poprawę ścieżki jest zapach jaki zostawiają mrówki.

Zapach nie jest niczym innym jak feromonem wytwarzanym przez mrówki. Dzięki pozostawionemu zapachowi mrówki wiedziały w jaki sposób poruszali się ich poprzednicy w związku z czym odtworzenie trasy nie stanowiło już poważnego problemu. Przy kolejnych iteracjach kolonia próbuje optymalizować aktualną trasę. W tym celu również wykorzystuje

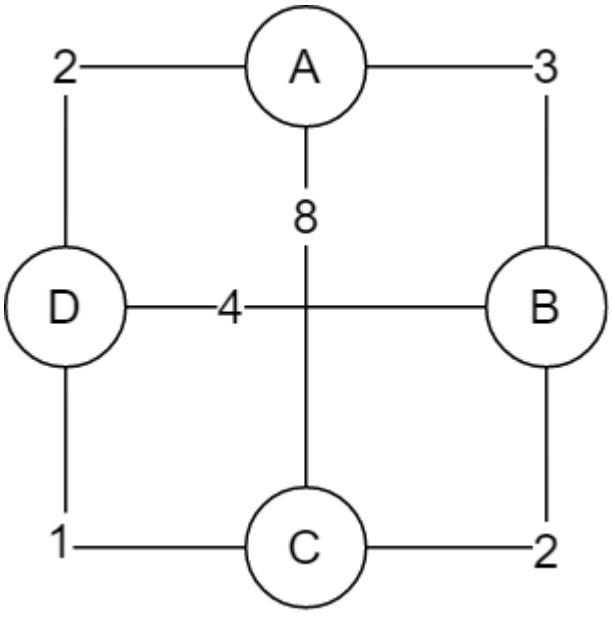
zapach pozostawiony w poprzednich przejściach. Ścieżka jest losowo zmieniana w celu optymalizacji. Jeśli modyfikacja przyniosła oczekiwany efekt, to trasa zostaje zmieniona.

Feromony są istotnym czynnikiem w całym procesie. To dzięki nim trasa jest ulepszana. Zapach posiada jedną z charakterystyk która na początku może wydawać się problematyczna. Wraz z upływem czasu siła zapachu słabnie aż do całkowitego zaniknięcia. Właściwość ta jest zaletę, a nie wadę. To dzięki pracy zespołowej zapach na najlepszej trasie jest podtrzymywany, a na słabszych zanika. Dzięki tej selekcji dłuższe trasy nie są brane pod uwagę w wyniku czego zostaje trasa najkorzystniejsza.

Do wyznaczenia optymalnej trasy potrzebne są długości jakie należy przebyć do przemieszczania się między punktami. W 1.1 przedstawione są przykładowe odległości.

Tabela 1.1: Wartości kosztów

	A	B	C	D
A	0	3	8	2
B	3	0	2	4
C	8	2	0	1
D	2	4	1	0

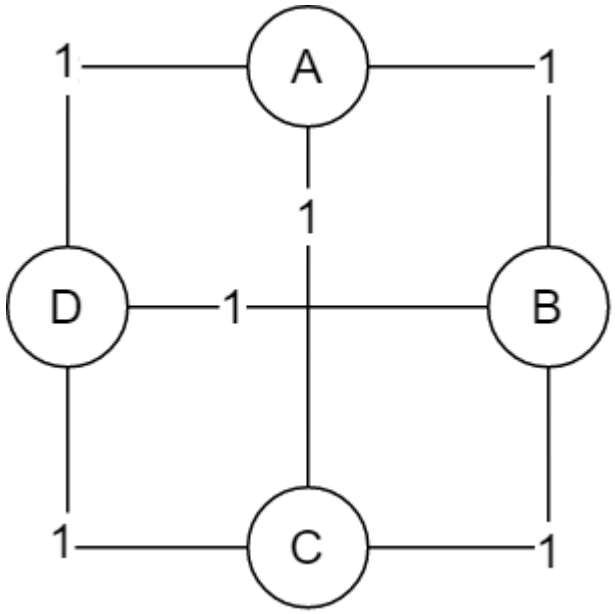


Rysunek 1.7: Graf z wagami

Równie ważne jest wyznaczenie początkowych współczynników feromonów. Na początku nadajmy wszystkim krawędziom w grafie wartości równe 1, a współczynnik parowania niech wynosi 0.5.

Tabela 1.2: Wartości feromonów

	A	B	C	D
A	0	1	1	1
B	1	0	1	1
C	1	1	0	1
D	1	1	1	0



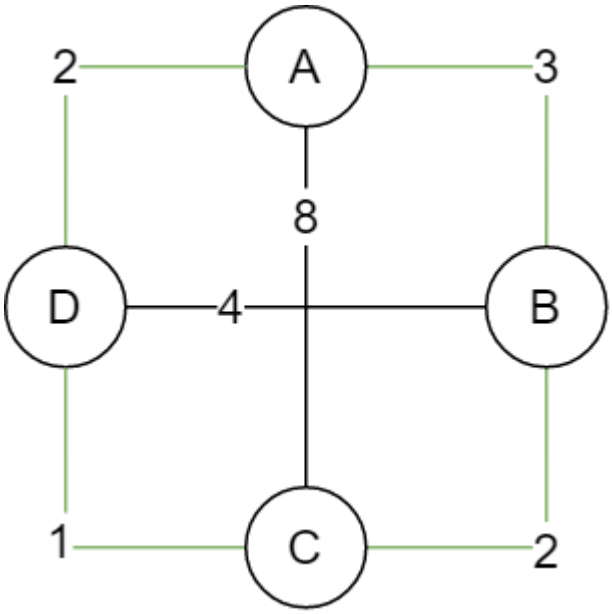
Rysunek 1.8: Graf z początkowymi wartościami feromonów

Posiadając początkowe dane wyznaczmy w sposób losowy trasy dla dwóch agentów: L1 i L2. Agent L1 poruszał się trasą w której odwiedził wierzchołki w następującej kolejności: A, B, C, D, A.

Agent L2 wyznaczył następującą trasę: A, C, B, D, A.

Mrówki odpowiednio pokonały dystans 8 i 16 punktów. Dzięki tej informacji można zaktualizować wartości feromonów na poszczególnych krawędziach. Do obliczeń wykorzystany jest wzór: TUTAJ WZÓR DODAĆ. Krawędzie A-B i C-D zostały odwiedzona jedynie przez agenta L1 w wyniku czego wartość feromonu zostaje zmieniona na 10/16. Następnie krawędzie A-C i B-D zostają zaktualizowane na 9/16. Krawędzie A-D i B-C są odwiedzone dwukrotnie a wartość feromonów wynosi 11/16.

Ostatnią fazą algorytmu jest wyznaczenie prawdopodobieństwa z jakim kolejni agenci będą wybierać kolejny wierzchołek. W kolejnej iteracji agent L3 znajduje się w wierzchołku B. Do wyboru ma krawędzie A, C i D. Według wzoru TUTAJ WZÓR DODAĆ wyliczane jest prawdopodobieństwo dla wszystkich możliwości. Przejście z krawędzi B do krawędzi



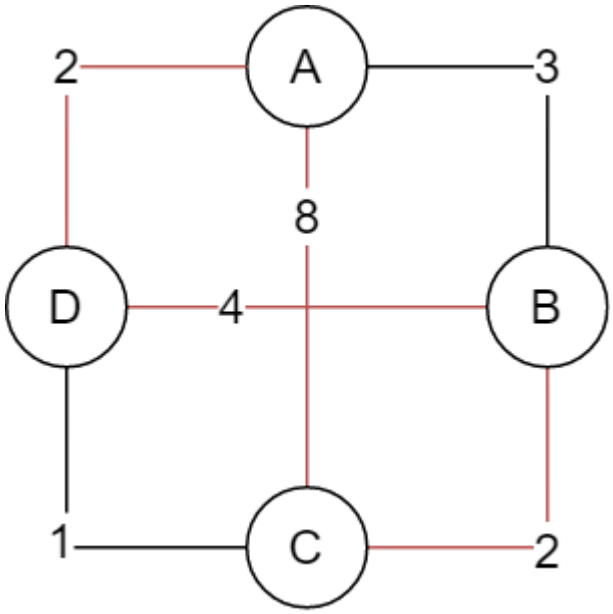
Rysunek 1.9: Trasa przebyta przez agenta L1

A wynosi około 20 PROCENT, do krawędzi D 30 procent, a do krawędzi C około 50 PROCENT.

Optymalna trasa zostanie wykształcona po wykonaniu wielu iteracji. Ilość iteracji nie jest zdefiniowana i dla każdego przypadku może być różna. Sytuacja wygląda identycznie w przypadku wyboru wartości p. Ważne jest natomiast to, aby w trakcie działania algorytmu nie modyfikować tej wartości. Powinno ona być taka sama na każdym kroku algorytmu.

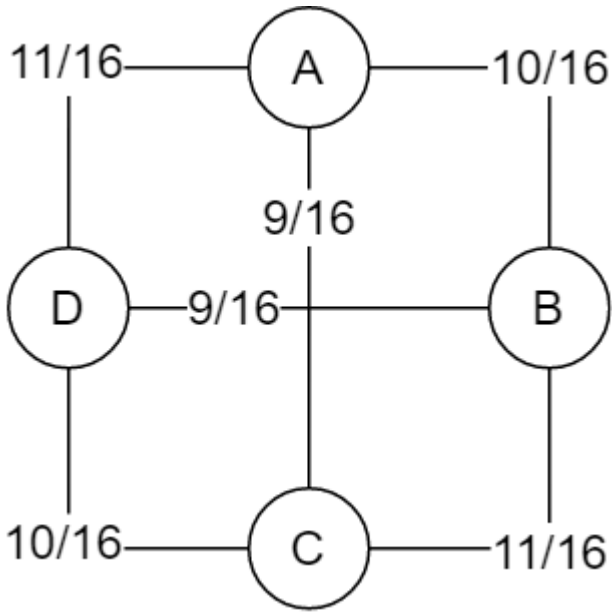
1.2.3 Algorytmy zachłanne

Kolejnym spojrzeniem na poruszany w pracy problem komiwojażera są algorytmy zachłanne (ang. greedy algorithms). Nie znajdziemy dowodu na to czy dla podanego problemu algorytm zachłanny znalazł poprawny wynik, jednak stosując się do pewnych zasad możemy określić, że dla naszego problemu istnieje rozwiązanie zachłanne. Główną strategią jaką się kierują jak sama nazwa wskazuje jest dokonywanie wyborów, które w danej chwili wydają się najlepsze. Oznacza to, że dokonuje się wyborów optymalnych lokalnie w nadziei, że te wybory doprowadzą do rozwiązania globalnego w rozsądnym czasie. W odróżnieniu do strategii zastosowanej w programowaniu dynamicznym wybory podejmowane przez algorytmy zachłanne nie są uzależnione od wyborów przeszłych. Kolejnym kryterium stosowanym do ocenienia poprawności rozwiązania zachłannego jest własność optymalnej pod struktury, mówiąca o tym, że optymalne rozwiązanie dla całego problemu istnieje



Rysunek 1.10: Trasa przebyta przez agenta L2

jedynie przy optymalnym rozwiązaniu pod problemów. Dane kryterium jest również istotne w przypadku rozwiązywania problemów metodą programowania dynamicznego. Algorytmy zachłanne nie zawsze prowadzą jednak do optymalnych rozwiązań, jednakże dla w większości problemów dają wystarczające rezultaty. Skorzystanie z algorytmów zachłannych często okazuje się niewystarczające. Aby uzyskać lepszy efekt i polepszyć zbudowane już trasy możemy wykorzystać algorytmy lokalnej optymalizacji (ang. local search). Użycie ich na zwróconych przez algorytmy zachłanne trasach powinno zminimalizować odległości między wierzchołkami w celu poprawienia otrzymanego rozwiązania. Dokładniejszy opis działania wybranych algorytmów zachłannych i metod optymalizujących został przedstawiony w podrozdziałach.



Rysunek 1.11: Graf z wartościami feromonów po modyfikacji

2. Algorytm genetyczny - Paweł

W tym rozdziale zostanie przedstawiony sposób w jaki zostanie zastosowany algorytm genetyczny przy wyznaczaniu najlepszych tras dla ciężarówek przewozu odpadów komunalnych. W przypadku mutacji oraz krzyżowania zostaną zbadane trzy różne metody, w celu znalezienia dającej najlepsze wyniki dla badanego problemu.

2.1 Chromosom

W algorytmie genetycznym, każdy osobnik z populacji reprezentuje jedno rozwiązanie. Jakość takiego rozwiązania jest zapisywana do zakodowanej postaci chromosomu. Chromosom z definicji jest to ciąg genów reprezentujący dane rozwiązanie. Z kolei gen przenosi informację o cechach. Możliwość osiągnięcia sukcesu w algorytmie genetycznym jest tylko wtedy, gdy odpowiednio zakoduje się cechy i ustali funkcję przystosowania. Do zakodowania badanego problemu zostanie użyta metody permutacyjna bez powtórzeń. Permutacja to jest dowolnie utworzony ciąg ze wszystkich elementów zbiorów. W badanym problemie każdemu punktowi przed wylosowanie tras zostanie przypisany unikalny indeks, będzie on odpowiadał genowi. Następnie dla każdego z N osobników zostanie zapisany chromosom w postaci ciągu permutacyjnego. Na rysunku 2.1 zostały zilustrowane przykłady kodowania permutacyjnego bez powtórzeń. W przyszłych podrozdziałach gen będzie oznaczał jeden z punktów załadowań na trasie ciężarówki. Chromosom będzie oznaczał kolejność odwiedzania tych punktów(genów) przez pojazd.

Rysunek 2.1: Kodowanie chromosomów

2.2 Funkcja przystosowania

Algorytm genetyczny szuka osobnika z największą wartością funkcji przystosowania. W badanym problemie należy znaleźć najkrótszą trasę. W momencie, gdy długość takiej trasy zostanie odwrócona, to okaże się, że im większa wartość odwrotności tym krótsza

trasa. Zatem wzór funkcji przystosowania to

$$fp = \frac{1}{s+1}$$

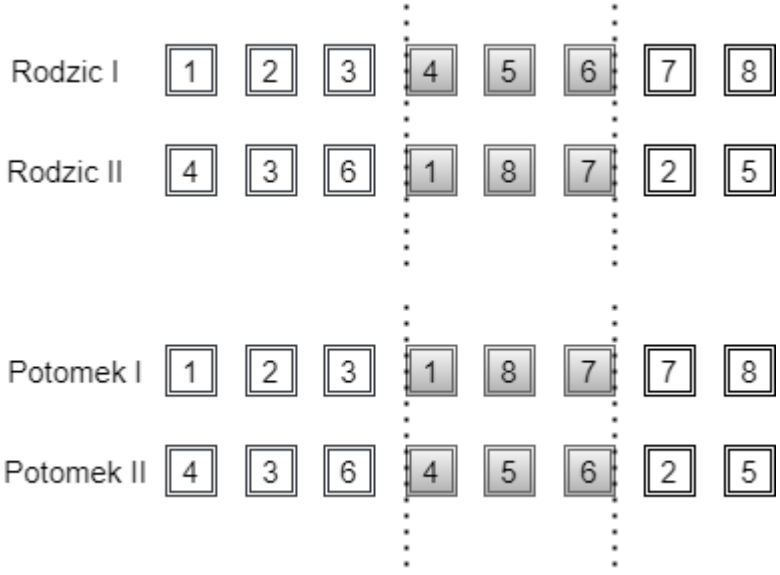
gdzie s - długość trasy, czyli suma odległości pomiędzy genami w chromosomie(1 - 2 - 3 - 4 - 5 - 6 - 7).

2.3 Metody krzyżowania

W pracy zostaną zbadane trzy rodzaje metod krzyżowania. Każde z nich charakteryzuje się czymś innym jeśli chodzi o liczbę potomków oraz porządek genów względem rodziców. Warunkiem koniecznym jest aby wynik krzyżowań, nie zaburzał struktury permutacyjnej chromosomu.

2.3.1 Krzyżowanie z częściowym odwzorowaniem - PMX

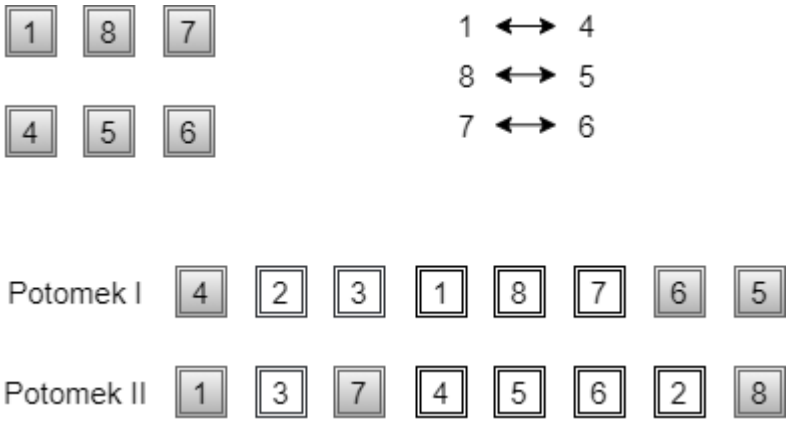
PMX(ang. partially mapped crossover) jest odmianą krzyżowania dwupunktowego, w którym powstaje dwójka potomstwa. Załóżmy, że mamy wyselekcjonowanych dwóch rodziców: 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 oraz 4 - 3 - 6 - 1 - 8 - 7 - 2 - 5 rys. 2.2.



Rysunek 2.2: Krzyżowanie PMX część 1

Losowane są dwa dowolne punkty w których się ich rozcina, w tym wypadku jest znajdują się one po trzecim i szóstym genie. Powstałe w ten sposób segmenty pomiędzy tymi punktami, są to: 4 - 5 - 6 oraz 1 - 8 - 7, wymienia się ze sobą. W wyniku tej operacji

powstały dwa chromosomy: 1 - 2 - 3 - 1 - 8 - 7 - 7 - 8 oraz 4 - 3 - 6 - 4 - 5 - 6 - 2 - 5. Oba z nich nie są permutacyjne i występują w nich powtórzenia. Należy je zamienić na te geny, które zostały wycięte. W tym celu następuje określenie relacji pomiędzy genami w wyciętych segmentach, które następnie się zamienia ze sobą w chromosomach(rys. 2.3).

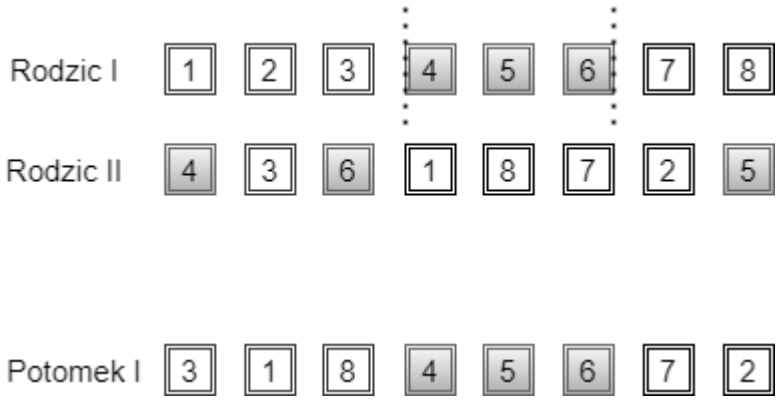


Rysunek 2.3: Krzyżowanie PMX część 2

W pierwszym dziecku zamieniamy: 1 -> 4, 8 -> 5 oraz 7 -> 6. W drugim należy wykonać odwrotne mapowanie: 4 -> 1, 5 -> 8 oraz 6 -> 7 Tak powstałe dzieci posiadają już strukturę permutacyjną i mogą brać udział w kolejnych etapach.

2.3.2 Krzyżowanie z zachowaniem porządku - OX

OX(ang. order crossover) jest również odmianą krzyżowania dwupunktowego. W przeciwieństwie do PMX wynikiem będzie tylko jedno dziecko. Rozważmy rodziców z poprzedniego przykładu. Pierwszym krokiem jest wylosowanie dwóch dowolnych punktów w których zostanie rozcięty pierwszy rodzic. Załóżmy podobnie jak poprzednio, że są to punkty po trzecim i szóstym genie(rys 2.4). Oba punkty tworzą segment 4 - 5 - 6.

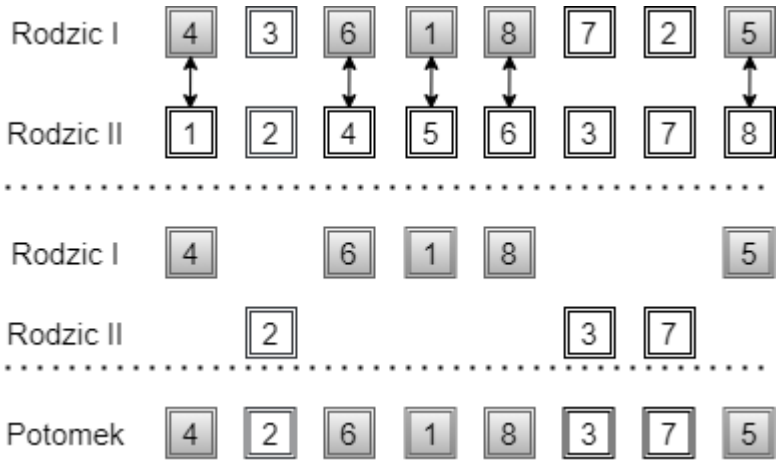


Rysunek 2.4: Krzyżowanie OX

Następnie w drugim rodzicu należy usunąć geny, które zostały z pierwszego rodzica wycięte. Na rysunku 2.4 zostały one podkreślone. Ostatnim krokiem jest wstawienie wycinka 4 - 5 - 6 do drugiego rodzica, w to samo miejsce z jakiego zostały usunięte. Powstał potomek 3 - 1 - 8 - 4 - 5 - 6 - 7 - 2. Na pierwszy rzut oka można zauważyć, geny które zostały wycięte z pierwszego rodzica znajdują się na tych samych miejscach, pozostałe natomiast się przemieściły.

2.3.3 Krzyżowanie cykliczne - CX

CX(ang. cycle crossover) w przeciwieństwie do PX i OX nie polega na krzyżowaniu w dwóch określonych punktach. Aby wyznaczyć potomka, należy w dowolnym rodzicu znaleźć cykl permutacji, zaczynając od dowolnego miejsca. Rozważmy dwa chromosomy: 4 - 3 - 6 - 1 - 8 - 7 - 2 - 5 oraz 1 - 2 - 4 - 5 - 6 - 3 - 7 - 8(rys. 2.5). Szukanie cyklu polega na kopiowaniu genów z jednego rodzica według pozycji określonych przez rodzica drugiego.



Rysunek 2.5: Krzyżowanie CX

Załóżmy, że zaczynamy od pierwszego genu czyli 4, jej odpowiednikiem w drugim rodzicu jest gen 1. Szukamy tego genu w pierwszym rodzicu. Znajduje się on na czwartym miejscu, jej odpowiednikiem jest gen 5, którego następnie szukamy w pierwszym chromosomie. Czynność powtarzamy do momentu, aż zostanie wykryty cykl. W podanym przykładzie cykl występuje dla 4 - 1 - 5 - 8 - 6. Odpowiednikiem dla 6 genu jest już 4, więc został wyznaczony cykl. Kolejnym krokiem jest wycięcie cyklu z chromosomu, w którym został wyznaczony. W ten sposób zostaje przepisane 4 - - 6 - 1 - 8 - - - 5. Aby skończyć krzyżowanie, w puste miejsce należy wpisać geny z drugiego rodzica, które nie wystąpiły w

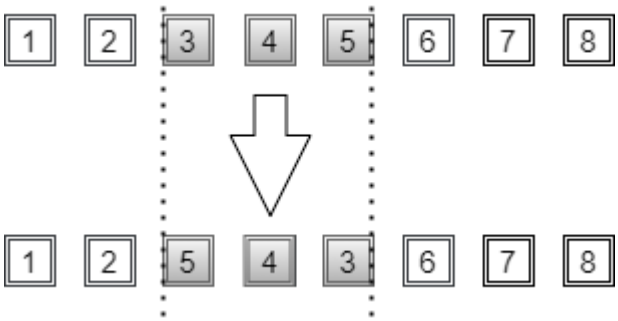
cyklu. W tak powstałym dziecku 4 - 2 - 6 - 1 - 8 - 3 - 7 - 5, wszystkie geny zajmują taką samą pozycję jak w którymś z rodziców, inaczej niż to było przy krzyżowaniu OX.

2.4 Mutacje

W pracy zostaną zbadane trzy różne rodzaje mutacji. Są to specjalne mutacje wykorzystywane przy strukturach permutacyjnych. Każda z nich zostanie również zbadana z różną wartością pm. Z reguły nie może być one duże, aby nie niszczyć potencjalnych rozwiązań. Powinno delikatnie wprowadzać różnorodność, aby była możliwość przeszukiwać nowe obszary.Poza tym bardzo ważne jest, aby zmiany powstałe w wyniku mutacji nie zaburzały struktury permutacyjnej chromosomu. Wszystkie mutacje zostaną opisane na tym samym chromosomie 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8.

2.4.1 Mutacja odwracająca

W mutacji odwracającej wybierany jest dowolny podciąg genów z chromosomu, a następnie ich kolejność jest odwracana. Załóżmy, że wybrany podciąg to 3 - 4 - 5(rys. 2.6). W tym przypadku składa się on z trzech genów, więc po odwróceniu zamienią się ze sobą dwa geny, środkowy zostanie na tym samym miejscu. Po mutacji końcowy chromosom ma postać 1 - 2 - 5 - 4 - 3 - 6 - 7 - 8. Struktura permutacyjna nie została zachwiana.

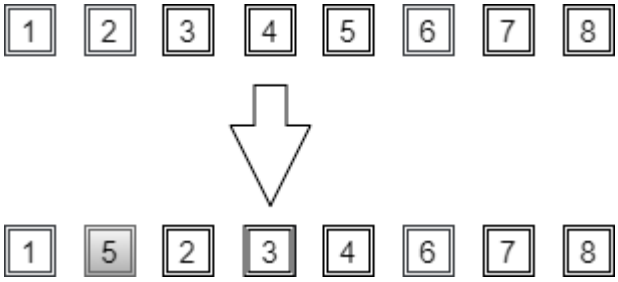


Rysunek 2.6: Mutacja odwracająca

2.4.2 Mutacja wstawiająca

W tej mutacji dowolny gen jest przestawia się w losowe miejsce. Jest to najprostsza mutacja, ale teoretycznie, może może generować rozwiązaniach w całkowicie nowych przestrzeniach przeszukiwań, gdyż jeśli punkt, który znajduje się na końcu trasy, może znaleźć się na początku. Na rysunku 2.7 został wylosowany gen 5, który znajdował się

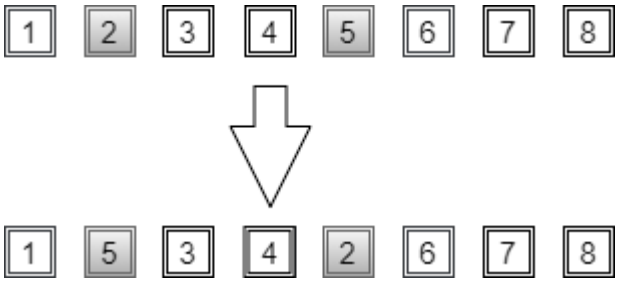
na piątym miejscu, po mutacji znalazł się na drugim miejscu, w rezultacie chromosom po mutacji ma postać 1 - 5 - 2 - 3 - 4 - 6 - 7 - 8.



Rysunek 2.7: Mutacja wstawiająca

2.4.3 Mutacja zamieniająca

W mutacji zamieniającej zamienia się dwa dowolne geny miejscami. Jest to tak naprawdę, rozszerzona wersja mutacji wstawiającej. Losujemy dwa dowolne geny, załóżmy, że są to 2 i 5, po czym zamieniamy je miejscami, tak jak na rysunku 2.8. Powstały chromosom to 1 - 5 - 3 - 4 - 2 - 6 - 7 - 8.



Rysunek 2.8: Mutacja wstawiająca

3. Algorytm mrówkowy

Słowo rozwój może być zestawiane z wieloma rzeczownikami. Wiele dziedzin ciągle się rozwija, powstają nowe udogodnienia które wpływają na wiele dziedzin życia. Dzięki rozwojowi techniki ludzie są w stanie osiągać cele które jakiś czas temu mogły być tylko marzeniami. Rozwój techniki również potrzebuje inspiracji do tworzenia nowych, lepszych rozwiązań

Szybkość oraz dokładność rozwoju zależy od wielu czynników. Dzięki pracy zespołowej pewne problemy mogą być rozwiązywane szybciej i dokładniej. Wysiłek włożony przez grupę procentuje szybko, a same efekty mogą być również satysfakcjonujące. Istotnym czynnikiem jest wysiłek wkładany przez każdego członka grupy.

Obserwując przyrodę możemy zauważyć w jaki sposób zwierzęta radzą sobie z różnymi problemami. Złożone grupy mogą być spokojniejsze o zdobycie pożywienia czy też o przetrwanie w ciężkich warunkach. Praca zespołowa jest jedną z cech której osobniki w grupie uczą się nie będąc nawet tego do końca świadomym.

3.1 Wprowadzenie do algorytmu

Na przestrzeni czasu wiele gatunków zwierząt żyjących na ziemi przystosowało się do panujących tu warunków. Jedną z takich gatunków są mrówki. Te niewielkich rozmiarów owady posiadają zdolności pomagające im przetrwać wśród najcięższych warunków. Mrówki żyją w stadach w związku z czym wykorzystują pracę zespołową do rozwiązywania problemów jakie codziennie napotykają na swojej drodze.

Aby zapewnić przetrwanie stadu mrówki potrzebują zapewnić sobie dostęp do pokarmu. Dziesiątki tysięcy mrówek mają swoje schronienie w mrowiskach. To tam trafia zdobyty przez nich pokarm. Wystarczy aby jedna mrówka znalazła miejsce z pokarmem, to po powrocie do mrowiska inne osobniki są w stanie odtworzyć drogę do pożywienia. Na tym etapie należałoby się zastanowić, w jaki sposób mrówki są w stanie komunikować się między sobą?

Jednym z opisywanych przez nas rozwiązań do wyznaczania zoptymalizowanej trasy jest algorytm mrówkowy, inaczej nazywany ACO - Ant Colony Optimization. Pomysł ten został zaczerpnięty z natury. Jak sama nazwa wskazuje działanie algorytmy jest związane z

mrówkami, a dokładnie z kolonią mrówek. Pomysł na algorytm został zaproponowany na początku lat 90 XX wieku przez włoskiego badacza - Marco Dorigo.

Tak jak wspomniałem wcześniej, algorytm opiera się na pracy mrówek. Chodzi tutaj dokładnie o trasę jako mrówki pokonują od swojego siedliska do miejsca w którym znajduje się pożywienie. Ważne jest znaczenie tutaj pracy zespołowej. Trasę kształtuje cała kolonia mrówek, a nie pojedyncze przypadki. Mrówki z każdą kolejną podróżą wykształcają coraz to bardziej optymalną trasę.

3.2 Opis działania algorytmu

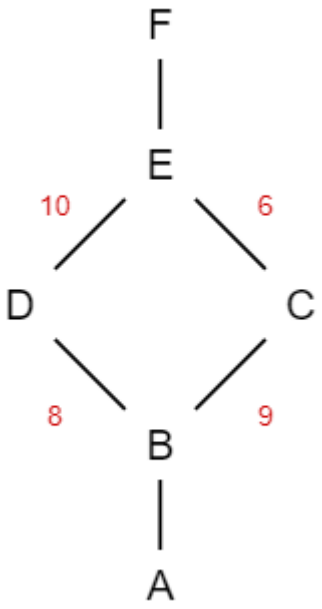
Mrówka w celu znalezienia pokarmu w sposób losowy wyrusza z mrowiska. Losowo przeszukując teren szuka pokarmu. Gdy już go znajdzie wraca do siedliska i informuje o tym fakcie pozostałe mrówki. Aby dostarczyć więcej pokarmu mrówki wyruszają do miejsca spoczynku pożywienia. Chcąc uniknąć sytuacji w której zdobycz może zostać zabrana przez inne owady, mrówki muszą jak najbardziej zoptymalizować trasę jaką mają do pokonania.

Mimo posiadania informacje o znalezionym pokarmie, każda mrówka sama musi zlokalizować źródło. Czy mrówki poruszają się tą samą trasą przy każdym wyjściu z mrowiska? Aby trafić do miejsca w którym znajduje się pokarm, wspomniane owady wykorzystują ślady pozostawione przez osobników które już natrafiły na pożywienie. W ten sposób mrówka która wyrusza w sposób losowy, natrafia na ślad poprzednika który jest wskazówką do znalezienia poszukiwanego pokarmu. Wspomniany ślad nazywa się feromonem. To dzięki tej właściwości mrówki są w stanie lokalizować trasy prowadzące do pokarmu.

Na rysunku **??** został przedstawiony graf z wierzchołkami A-B-C-D-E-F. Nad krawędziami kolorem czerwonym zostały oznaczone wagi. Na początku załóżmy, że mamy do dyspozycji 80 agentów. Przez kilka pierwszych iteracjach mrówki poruszały się losowo i powstał następujący podział:

Tak jak można to zauważyć na grafice **??**, po pierwszych iteracjach, przez obie ścieżki przechodzi taka sama ilość agentów. Dzieje się tak ponieważ mrówki rozpoczynają pracę w sposób losowy. W dalszych krokach następują modyfikacje i agenci dążą do wyznaczenia najoptymalniejszej ścieżki.

Ilość agentów odwiedzających ścieżki zmienia się. Bardziej optymalna trasa zyskuje widoczną przewagę. W kolejnych iteracjach mrówki wykorzystują siłę feromonów. Bardziej



Rysunek 3.1: Początkowy graf

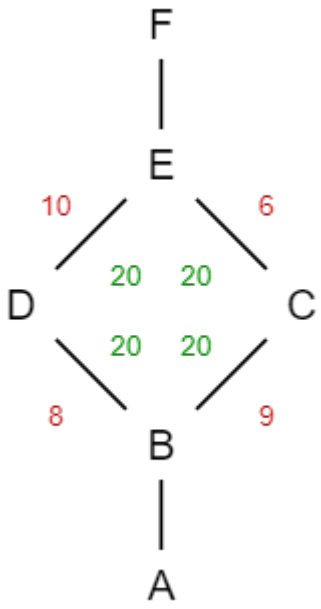
optymalne ścieżki są częściej odwiedzane w związku z czym zapach na tych krawędziach jest silniejszy oraz podtrzymywany. Na mniej optymalnych trasach zapach zanika i przestają one być atrakcyjne dla agentów. Opisana sytuacja prowadzi do wyznaczenia trasy która jest najatrakcyjniejsza do przebycia dla mrówek.

Feromony posiadają cechę która może się wydawać, że negatywnie wpływa na wyznaczanie ścieżki. Chodzi tutaj o ulatnianie się zostawionego zapachu. Na pierwszy rzut oka może się to zjawisko wydawać niepożądanym, ale w rzeczywistości ma duży wpływa na optymalizację. Jeśli feromony nie straciłyby na swojej sile, to bardzo prawdopodobne, że pierwotna ścieżka mogłaby zostać uznana za najbardziej optymalną.

W jaki sposób wyznaczona zostaje najbardziej optymalna ścieżka? Zapach feromonów jest podtrzymywany przez wędrujące mrówki. Z czasem owady te same zbaczają z drogi w celu poszukiwania alternatywnej trasy. Jeśli wybrana trasa jest optymalniejsza od pozostałych to ślad jest podtrzymywany, a na innych zanika. Dzięki temu w sposób iteracyjny można wyróżnić trasę najoptymalniejszą, a słabsze z czasem zostają odrzucone ponieważ przestają być odwiedzane.

3.3 Pozostałe zastosowania algorytmu mrówkowego

Algorytm mrówkowy znajduje swoje zastosowanie w rozwiązywaniu innych problemów. Problem plecakowy jest jednym z takich przykładów. Pojawia się on najczęściej przy

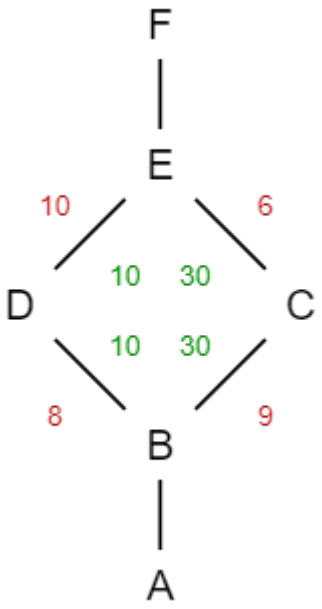


Rysunek 3.2: Rozkład ścieżek na grafie po pierwszych iteracjach

optymalnym zarządzaniu zasobami. Mamy dany zbiór jakichś przedmiotów z czego każdy z nich posiada określony ciężar i wartość. Do plecaka musimy załadować przedmioty o jak największej wartości. Naszym ograniczeniem jest jednak łączny ciężar przedmiotów które możemy udźwignąć.

Algorytm mrówkowy wygląda bardzo podobnie w tym przypadku. Na początku mamy pewnę geracje agentów - mrówek. Każdy agent iteracyjnie poszukuje jak najlepszego rozwiązania. Po każdej iteracji można wyróżnić trzy typy rozwiązań: rozwiązanie pośrednie, rozwiązanie częściowe lub stan. Agenci w celu znalezienia rozwiązania wykorzystuje swoje naturalne umiejętności czyli zostawia na wszystkich obiektach w plecaku feromony. Dzięki lotności feromonów mrówki są w stanie identyfikować bardziej zadowalające przedmioty.

Krzysztof Schiff w jednym z artykułów przedstawił rozwiązanie problemu plecakowego przy użyciu trzech metod optymalizacji algorytmu mrówkowego. Wszystkie te metody można opisać za pomocą wzorów. Zgodnie z przyjętą konwencją przez autora artykułu metody mają odpowiednie nazwy: AKA1, AKA2 oraz AKA3. Atrakcyjność jest obliczana według wzorów: AKA1 = z/ w/V, gdzie: - z jest zyskiem wybranego obiektu; - w jest wagą wybranego obiektu; - V jest aktualną ładownością plecaka AKA2 = z/ $\hat{w2}$, gdzie: - z jest zyskiem wybranego obiektu; - w jest wagą wybranego obiektu; AKA3 = z/ w/C, gdzie: - z jest zyskiem wybranego obiektu; - w jest wagą wybranego obiektu; - C jest całkowitą wagą plecaka



Rysunek 3.3: Rozkład agentów w grafie po optymalizacji

Poniżej przedstawione zostały wyniki badań:

WYNIKI BADAN

Problem komiwojażera i problem plecakowy są problemami kombinatorycznymi i ich rozwiązanie polega poszukiwaniu optymalnej ścieżki na grafie pełnym. Inną wariacją problemu jest poszukiwanie rozwiązania problemu od razu zadanego na grafie. Jako przykład może posłużyć problem kolorowania grafu. Dla wszystkich wierzchołków w grafie należy dobrać takie kolory, aby żadne dwa sąsiednie wierzchołki nie miały tego samego koloru.

Mrówki nie działają bezpośrednio na grafie początkowym ponieważ graf ten nie musi być grafem pełnym. Należy stworzyć dla mrówek alternatywę podobną do oryginału z zachowaniem takiego samego zbioru wierzchołków ale z pełnymi krawędziami. Następnie należy dobrać numeryczne wartości odpowiadające konkretnym kolorom. Jeśli mrówka odwiedzi dany wierzchołek, to zostaje on pokolorowany na najniższy kolor który nie został dotychczas użyty do kolorowania któregoś z sąsiadów.

Tak jak w przypadku poprzednich algorytmów wykorzystywane są zapachy pozostawiane przez mrówki. Ilość użytych unikalnych kolorów byłaby odwrotnie proporcjonalna do ilości feromonów. W efekcie czego heurystyka byłaby odwrotnie proporcjonalna do wykorzystanych kolorów po kolejnych iteracjach. Wynikiem tych operacji będzie rozwiązanie w którym w grafie oryginalnym każdy wierzchołek będzie odwiedzany tylko raz.

Ostatni z przykładów wykorzystania algorytmu mrówkowego jest harmonogram produkcji. W porównaniu do poprzednich metod w tym algorytmie zachodzi pewna modyfikacja. Głównym problemem w harmonogramie produkcji jest znalezienie takiej kolejności przetwarzanych zadań, aby jak najszybciej je przetworzyć. Aby lepiej zobrazować tą sytuację należy sobie wyobrazić fabrykę w której znajdują się linie produkcyjne. Na linii są przetwarzane zadania w odpowiedniej kolejności oraz każde z zadań może zostać wykonane w różnym czasie.

W tej metodzie, podobnie jak w metodzie do rozwiązania problemu plecakowego, należy stworzyć graf pełny z wierzchołkami odpowiadającymi konkretnym zadaniom. Następnie mrówki przechodzą przez wszystkie wierzchołki i zostawiają feromony. Czynnikiem decydującym o wyborze wierzchołków nadal jest związana z feromonami. Do rozwiązania tego problemu nie jest brana pod uwagę ilość feromonów na krawędzi pomiędzy wierzchołkiem a jego sąsiadami. Wykorzystywana jest natomiast suma feromonów na wszystkich krawędziach do odwiedzanych wierzchołków z wierzchołkami już odwiedzonymi.

4. Algorytmy zachłanne

Istnieje wiele algorytmów zachłannych pozwalających otrzymać optymalne rozwiązanie dla problemu znalezienia najkrótszej trasy. W tym rozdziale skupimy się jednak na algorytmie najbliższego sąsiada, algorytmie najmniejszej krawędzi oraz algorytmie a*. W celu zoptymalizowania otrzymanych rozwiązań stosować będziemy operator 2-opt oraz operator 3-opt.

4.1 Algorytm najbliższego sąsiada

Poniższy podrozdział przybliży działanie algorytmu najbliższego sąsiada. Wszystkie wymagane dla algorytmu operacje zostaną opisane krok po kroku oraz przedstawione na krótkim przykładzie.

4.1.1 Wprowadzenie do algorytmu

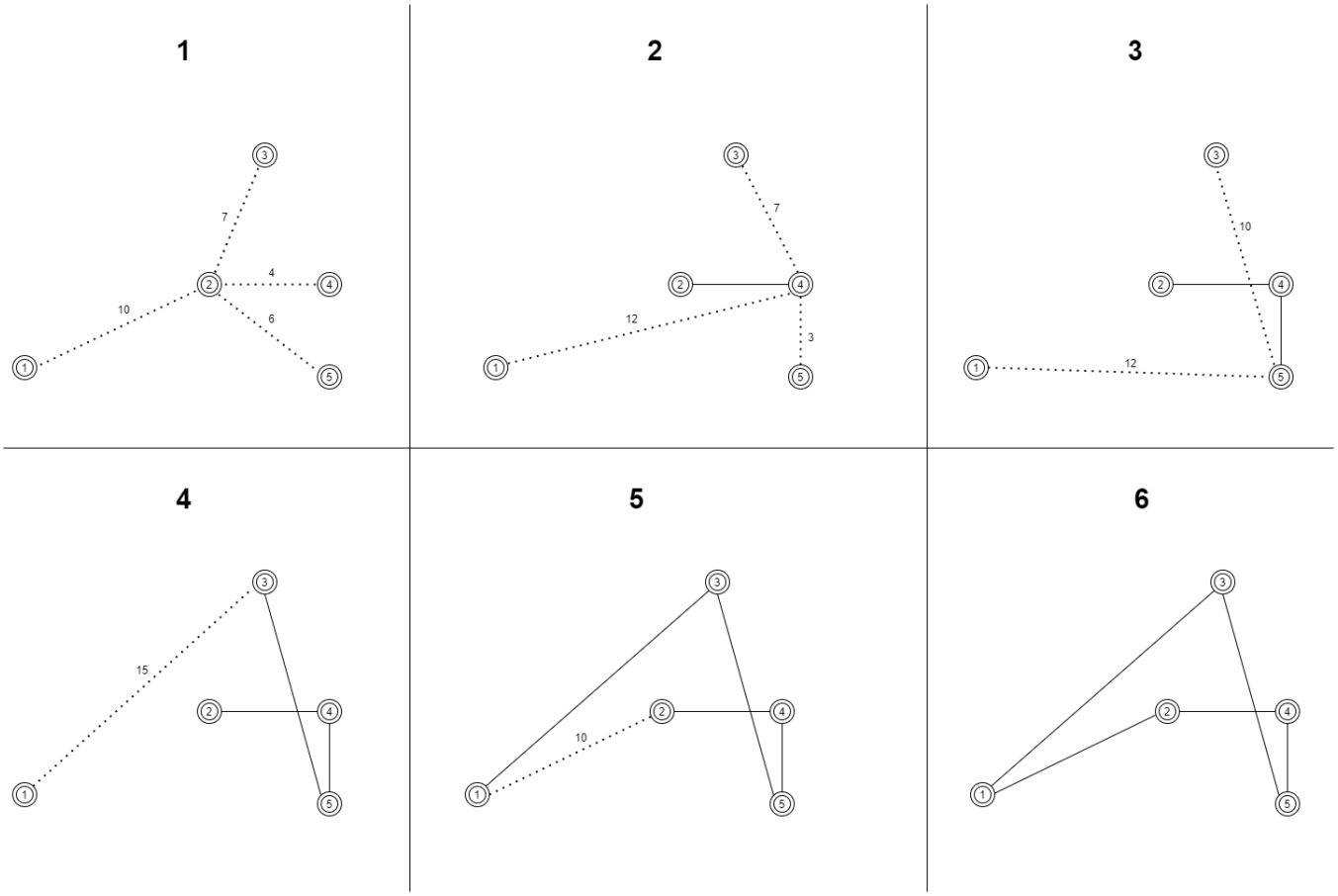
Jak sama nazwa wskazuje jest to algorytm polegający na odwiedzaniu, zaczynając od wybranego wierzchołka początkowego następnego wierzchołka znajdującego się najbliżej poprzednio odwiedzonego. Omawiany algorytm możemy podzielić na następujące kroki:

- 1 Wybieramy wierzchołek początkowy, który staje się naszym aktualnym i oznaczamy go jako odwiedzony.
- 2 Następnie dla aktualnego wierzchołka obliczamy i wyznaczamy najkrótszą krawędź, która połączy nasz aktualny wierzchołek tylko z tymi, które są nieodwiedzone.
- 3 Znalezioną najkrótszą krawędź w punkcie nr 2 dołączamy do rozwiązania.
- 4 Drugi wierzchołek, który należy do znalezionej krawędzi staje się naszym aktualnym oraz oznaczamy go jako odwiedzony.
- 5 Jeżeli w naszym grafie znajdują się jeszcze nieodwiedzone wierzchołki, przechodzimy do punktu 2.
- 6 Natomiast jeżeli wszystkie wierzchołki są już odwiedzone, łączymy ostatni aktualny z początkowym wierzchołkiem tworząc cykl oraz kończąc działanie algorytmu.

Po wykonaniu wszystkich kroków otrzymujemy optymalne rozwiązanie dla algorytmu najbliższego sąsiada.

4.1.2 Opis działania algorytmu

Aby lepiej zobrazować otrzymanie rozwiązania przez algorytm prześledźmy jego działanie na przykładzie przedstawionym poniżej na rysunku 5.1 .



Rysunek 4.1: Algorytm najbliższego sąsiada

Zaczynamy od wyboru wierzchołka nr 2 jako początkowego i tym samym ustawiamy go jako aktualny w danym momencie. Po obliczeniu wszystkich odległości prowadzonych do wierzchołków nieodwiedzonych wychodzi, że najkrótsza krawędź prowadzi do wierzchołka nr 4, więc dołączamy wybrana krawędź do rozwiązania i ustawiamy nowy aktualny wierzchołek, który staje się również odwiedzonym. W naszym grafie znajdują się nadal nieodwiedzone wierzchołki, dlatego powtarzamy krok algorytmu w celu znalezienia kolejnej najkrótszej krawędzi. Kolejną najlepszą odnalezioną w danym momencie krawędzią, którą dodamy do naszego rozwiązania będzie krawędź o wartości 3 łącząca nasz aktualny

wierzchołek z wierzchołkiem nr 5. Kroki 4, 5 oraz 6 przedstawiają kolejne powtarzalne iteracje algorytmu dochodząc w ostatnim kroku do utworzenia cyklu i tym zakończeniu działania algorytmu. W ten sposób otrzymaliśmy zachłanne rozwiązanie 2 - 4 - 5 - 3 - 1 - 2. Warto zaznaczyć, że dzięki oznaczaniu wierzchołków jako odwiedzone nie musimy w żadnej iteracji martwić się o to czy dołączenie kolejnej krawędzi z nieodwiedzonym wierzchołkiem spowoduje utworzenie niepożądanego cyklu.

4.2 Algorytm najmniejszej krawędzi

Kolejny podrozdział algorytmów zachłannych został poświęcony dokładniejszemu opisie działania algorytmu najmniejszej krawędzi, który w swoim działaniu przypomina algorytm poszukujący minimalnego drzewa rozpinającego, poprzez dołączenie do aktualnego rozwiązania najkrótszych wśród dopuszczalnych krawędzi.

4.2.1 Wprowadzenie do algorytmu

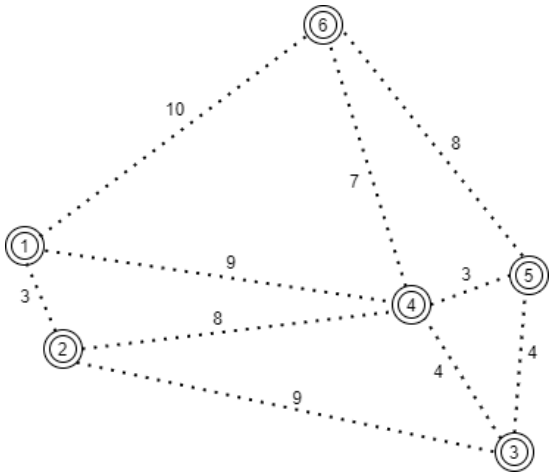
Aby otrzymać zachłanne rozwiązanie przy wykorzystaniu algorytmu najmniejszej krawędzi należy wykonać następujące kroki:

- 1 Algorytm zaczynamy od posortowania wszystkich możliwych krawędzi rosnąco według ich wag oraz na umieszczeniu ich w kolekcji.
- 2 Następnie z podanej kolekcji wybieramy krawędź o najmniejsze wadze oraz usuwamy ją z naszej kolekcji.
- 3 Jeśli dołączenie wybranej aktualnie krawędzi nie spowoduje utworzenia cyklu oraz utworzenia wierzchołka o trzech krawędziach to możemy dołączyć ową krawędź do rozwiązania. (pomijamy warunki, jeśli jest to ostatnia iteracja)
- 4 Gdy liczba wszystkich wierzchołków równa się liczbie dołączonych krawędzi do rozwiązania, oznacza to, że został utworzony cykl i należy zakończyć działanie algorytmu.
- 5 W przeciwnym wypadku wracamy do punktu nr 2.

Po zakończeniu działania algorytmu otrzymujemy optymalne rozwiązanie dla algorytmu najmniejszej krawędzi.

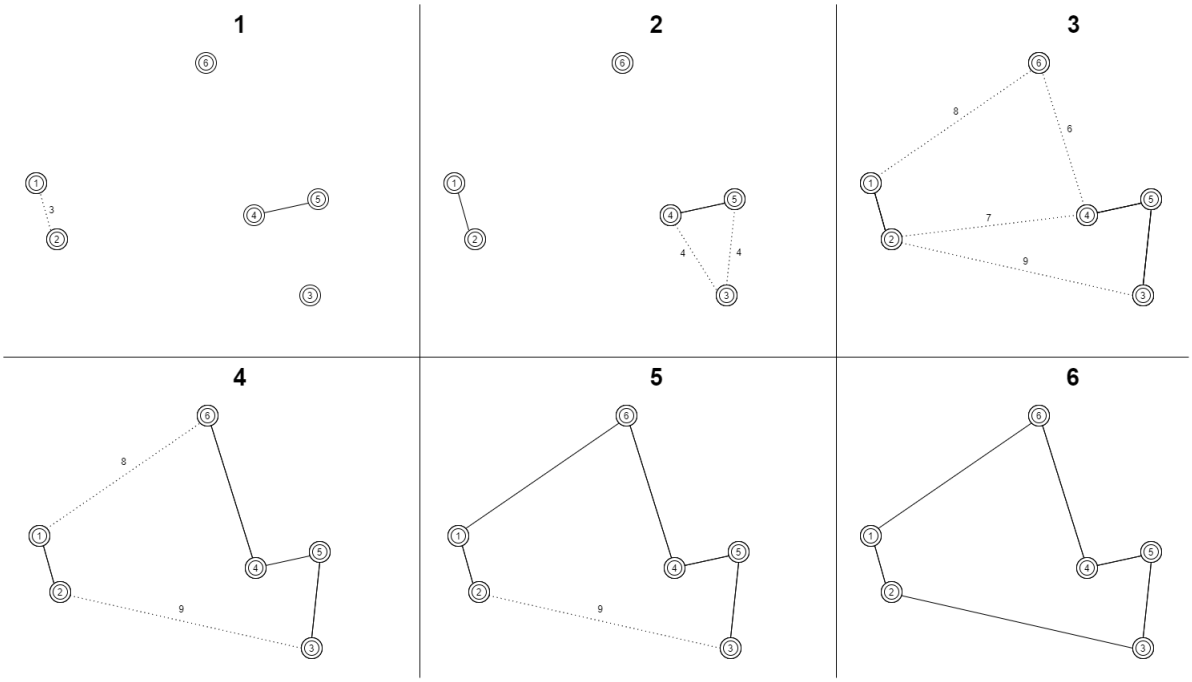
4.2.2 Opis działania algorytmu

Działanie algorytmu przedstawione zostanie na rysunku 4.2 przedstawiającym sześć losowo rozmieszczonych wierzchołków.



Rysunek 4.2: Początkowe rozmieszczenie wierzchołków

Natomiast wizualizacja kolejnych kroków algorytmu została przedstawiona na rysunku 5.3. Po posortowaniu wszystkich dostępnych krawędzi dla każdego wierzchołka rozpoczynamy działanie algorytmu.



Rysunek 4.3: Algorytm najmniejszej krawędzi

Podczas pierwszej iteracji okazuje się, że mamy aktualnie dwie krawędzie z najmniejszą wagą o wartości 3 (1 - 2 oraz 4 - 5), nie ma więc znaczenia, która krawędź dodamy

w pierwszej kolejności, ponieważ żadna z wybranych nie utworzy nam w tym momencie cyklu. W przypadku drugiej iteracji wybieramy krawędź o najmniejszej wadze, dołączając ją do aktualnego rozwiązania. Analogiczna sytuacja do pierwszej iteracji występuje w trzeciej iteracji, gdzie nie ma różnicy, która krawędź dołączymy do rozwiązania. Czwarta iteracja pokazuje sytuacje, w której nie możemy połączyć krawędzi 3 - 4, ponieważ utworzyłoby to niedozwolony w trakcie algorytmu cykl, dlatego tym razem wybieramy inne połączenie o najmniejszej wadze (4 - 6). Kolejny krok przedstawia przedstawia przypadek, gdzie nie jest możliwe połączenie krawędzi 2 - 4 (najmniejsza wartość - 7), ponieważ spowodowałoby dla wierzchołka nr 4 utworzenie trzech wychodzących z niego krawędzi, więc do rozwiązania dochodzi kolejna najmniejsza krawędź 1 - 6. W ostatnim kroku, pomimo dostępnych krawędzi z mniejszą wagą wybieramy krawędź 2 - 3, ponieważ tylko ona spowoduje utworzenie kompletnego cyklu. W takim wypadku kończąc działanie algorytmu otrzymujemy rozwiązanie 1 - 6 - 4 - 5 - 3 - 2 - 1.

4.3 Algorytm A*

Ostatnim omówionym rozwiązaniem do znajdowania najkrótszej ścieżki w grafie jest algorytm a*, w którym zawsze zostanie znalezione najlepsze zachłanne rozwiązanie. Metoda jest przykładem wybierania rozwiązań w danym momencie najlepszych, ale gwarantuje, że każdy wierzchołek zostanie odwiedzony. Najlepsze wyniki możemy otrzymać, wtedy kiedy obszar przeszukiwań jest strukturą drzewiastą.

4.3.1 Wprowadzenie do algorytmu

Główną zasadą algorytmu a* jest minimalizacja funkcji kosztu oraz funkcji heurystycznej, gdzie ta ostatnia musi spełniać dwa wymagane warunki tj. warunek dopuszczalności i warunek monotoniczności. Warunek dopuszczalności polega na tym, aby funkcja heurystyczna nie oszacowywała, gdy minimalizujemy koszt, a przeszacowywała, gdy chcemy zmaksymalizować zysk. Natomiast warunek monotoniczności mówi o tym, że im bliżej jesteśmy rozwiązania, tym oszacowywanie wyniku musi być coraz mniej optymistyczne.

4.3.2 Opis działania algorytmu

Jeśli przestrzeń przeszukiwań zawierać będzie ścieżki to możemy wówczas sprowadzić problem do problemu poszukiwania najkrótszej ścieżki w grafie. Wiemy, że musimy stworzyć ścieżkę zawierającą krawędzi, oznaczmy więc jako liczbę krawędzi, które już wykorzystaliśmy. Wtedy funkcją heurystyczną może być na przykład iloczyn liczby krawędzi pozostałych do wykorzystania i wagi najtańszej krawędzi – wtedy taka funkcja na pewno będzie nadmiernie optymistyczna. Dokładniejsze oszacowanie daje nam funkcja heurystyczna zdefiniowana jako suma najmniejszych wag niewykorzystanych jeszcze krawędzi. <tu jeszcze pojawią się rysunki i opis>

4.4 Optymalizacja otrzymanych rozwiązań

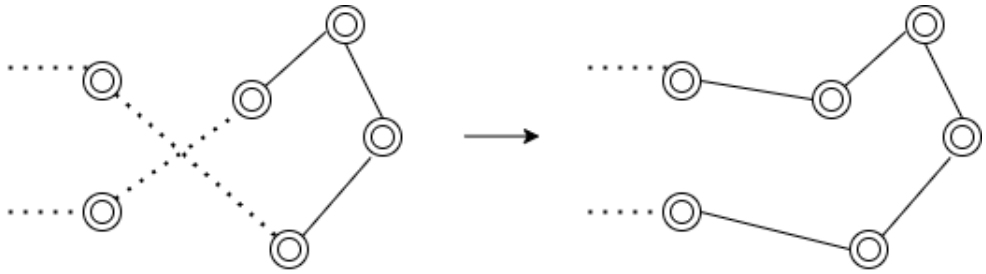
W ostatnim podrozdziale skupimy się na sposobach optymalizacji otrzymanych przez algorytmy zachłanne rozwiązań. Dokładniej omówiona zostanie metoda 2-opt oraz metoda 3-opt.

4.4.1 Metoda 2-opt

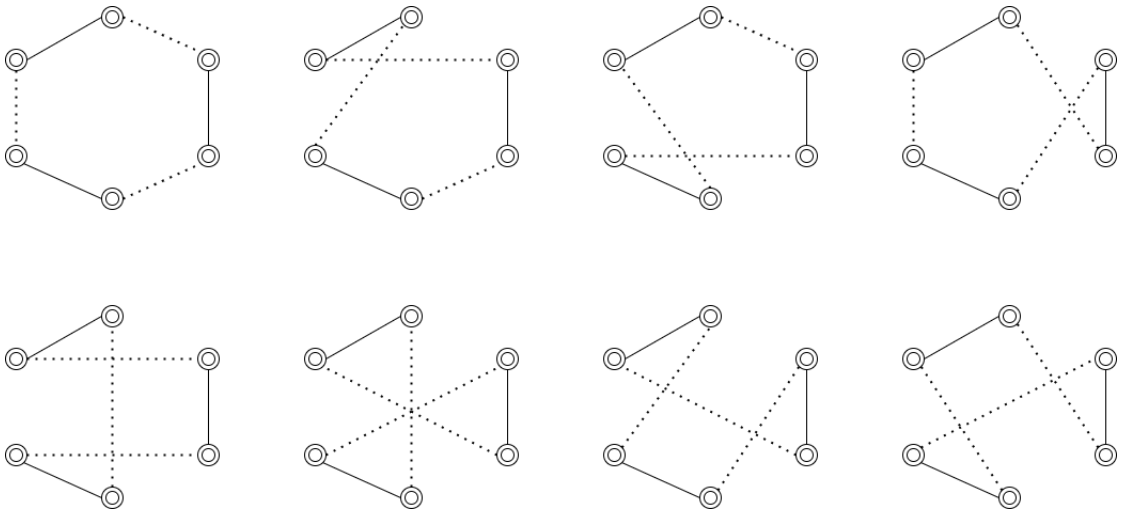
Metoda optymalizacyjna 2-opt polega na pozbyciu się z cyklu dwóch krawędzi w celu zastąpienia ich innymi krawędziami w taki sposób, aby otworzyć zupełnie inny cykl. Iteracje możemy powtarzać dla każdej pary krawędzi, oprócz tych sąsiadujących ze sobą, ponieważ ich zamienienie nie przyniesie żadnej modyfikacji. Metoda nie ma na celu zmiany położenia wierzchołków, jedynie kolejności ich odwiedzania. Po wykonaniu całej optymalizacji, należy sprawdzić która modyfikacja przyniosła najlepszy efekt skrócenia długości cyklu. W przypadku jeżeli żadna modyfikacja nie dała lepszego rozwiązania, nie modyfikujemy rozwiązania. Algorytm można wykonywać wielokrotnie, w ten sposób zostanie zrealizowane minimum lokalne. Dla lepszego przedstawienia działania metody 2-opt, należy spojrzeć na rysunek 5.4:

4.4.2 Metoda 3-opt

W odróżnieniu do algorytmu 2-opt, w metodzie 3-opt rozważane są wszystkie cykle, które możemy uzyskać wymieniając trzy krawędzie z cyklu aktualnego, przy czym możemy to wykonać na wiele sposobów tak jak zostało to przedstawione na rysunku 5.5.



Rysunek 4.4: Metoda 2-opt



Rysunek 4.5: Metoda 3-opt

Prosta tabela 4.1.

Tabela 4.1: Długi podpis tabeli 1, który pojawi się nad nią. Jak chcecie podpis pod tabelą, umieśćcie caption przed samym end{table} - ale to niezgodne z wytycznymi.

Kolumna 1	Kolumna 2	Kolumna 3	Kolumna 4
Kolumna 1	Kolumna 2	Kolumna 3	Kolumna 4
Kolumna 1	Kolumna 2	Kolumna 3	Kolumna 4
Kolumna 1	Kolumna 2	Kolumna 3	Kolumna 4

Przykładowa tabela 4.2, nieco bardziej skomplikowana.

Tabela 4.2: Długi podpis tabeli 2, który pojawi się nad nią

Kolumna wyróżniona	Kolumna pierwsza	Kolumna druga	Kolumna kolejna długa nazwa	Przeno-szenie słowa	Kolumna kolejna	Kolumna kolejna	Kolumna kolejna	Kolumna kolejna	Kolumna kolejna	Kolumna kolejna
Wiersz jakiś tam	1	2	3	4	5	6	7	8	9	10
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	**92,38**	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	**92,38**	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	**41,92**	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	**29,61**	55,02	72,33	95,82
Wiersz ze statystykami	11,56	92,38	827,21	41,92	29,71	28,77	29,61	55,02	72,33	95,82

5. Badania

5.1 Wyniki algorytmu genetycznego

5.2 Wyniki algorytmu mrówkowego

5.3 Wyniki algorytmów zachłannych

6. Porównania wyników

Podsumowanie

Tutaj będzie podsumowanie.

Spis tabel

Spis rysunków

Spis listingów

Spis algorytmów