Politechnika Śląska
Wydział Matematyki Stosowanej
Kierunek Informatyka
Studia stacjonarne I stopnia

Projekt inżynierski

Porównanie wybranych algorytmów heurystycznych w rozwiązywaniu zagadnień odwrotnych

Kierujący projektem:
dr inż. Adam Zielonka

Autor:
Kamil Kryus

Gliwice 2018

Projekt inżynierski:

Porównanie wybranych algorytmów heurystycznych w rozwiązywaniu zagadnień odwrotnych

kierujący projektem: dr inż. Adam Zielonka

autor: Kamil Kryus

Podpis autora projektu

. . . . . .

Podpis kierującego projektem

. . . . . .

Oświadczenie kierującego projektem inżynierskim

Potwierdzam, że niniejszy projekt został przygotowany pod moim kierunkiem i kwalifikuje się do przedstawienia go w postępowaniu o nadanie tytułu zawodowego: inżynier.

Data Podpis kierującego projektem

Oświadczenie autora

Świadomy/a odpowiedzialności karnej oświadczam, że przedkładany projekt inżynierski na temat:
Porównanie wybranych algorytmów heurystycznych w rozwiązywaniu zagadnień odwrotnych
został napisany przeze mnie samodzielnie.
Jednocześnie oświadczam, że ww. projekt:

- nie narusza praw autorskich w rozumieniu ustawy z dnia 4 lutego 1994 roku o prawie autorskim i prawach pokrewnych (Dz.U. z 2000 r. Nr 80, poz. 904, z późn. zm.) oraz dóbr osobistych chronionych prawem cywilnym, a także nie zawiera danych i informacji, które uzyskałem/am w sposób niedozwolony,
- nie była wcześniej podstawą żadnej innej urzędowej procedury związanej z nadawaniem dyplomów wyższej uczelni lub tytułów zawodowych.
- nie zawiera fragmentów dokumentów kopiowanych z innych źródeł bez wyraźnego zaznaczenia i podania źródła.

Podpis autora projektu

Kamil Kryus, nr albumu:246591, ... (podpis:) .....

Gliwice, dnia .....

Spis treści

Wstęp **7**

1. Opis problemu **9**

1.1. Tytuł podrozdziału 9

1.2. Cel 9

2. Opis algorytmów **11**

2.1. Algorytm symulowanego wyżarzania 11

2.1.1. Parametry 11

2.1.2. Kroki algorytmu 13

3. Funkcje testowe **15**

3.1. Funkcja kwadratowa dwóch parametrów 15

3.2. Funkcja Rastrigina 15

3.3. Funkcja Rosenbrocka 15

4. Dobór parametrów **17**

5. Implementacja **19**

6. Dostosowanie algorytmów do funkcji testowej zadań odwrotnych **21**

7. Narzędzia i technologie **23**

7.1. Użyte narzędzia 23

7.1.1. System kontroli wersji 23

7.1.2. Podział pracy na mniejsze zadania 23

7.2. Użyte technologie 23

8. Podsumowanie **25**

8.1. Dalsze kierunki rozwoju 25

8.2. Źródła 25

9. Tytuł drugiego rozdziału. Bardzo długi tytuł. Jego formatowanie jest trudniejsze **27**

Dodatek: Mój specjalny dodatek **29**

Rysunki **31**

Programy **33**

Literatura **35**

Wstęp

Na problem znalezienia optymalnego rozwiązania możemy trafić w wielu dziedzinach życia i nauki, np. w matematyce szukając globalnego minimum/maximum lub szukając najkrótszego połączenia pomiędzy kilkoma miastami (problem komiwojażera). Szukając rozwiązań, zawsze chcemy by rozwiązanie było jak najlepsze (lub dokładnie najlepsze) i zostało znalezione w rozsądnym czasie. W tym celu właśnie jest używana heurystyka.

Metody heurystyczne znacznie skracają nam czas wyszukiwania rozwiązania problemu, aczkolwiek często ich wynikiem jest jedynie wynik bardzo zbliżony do najlepszego. Pozwala jednak nam to na jedną z dwóch rzeczy:

1. zaakceptowanie takiego wyniku, gdy dokładne rozwiązanie nie jest konieczne (np. kompresja obrazu),
2. zawężenia zakresu i dalszych poszukiwań najlepszego rozwiązania.

Jednak aby metodę heurystyczną uznać za dobrą, musi ona spełnić 3 wymagania:

- rozwiązanie jest możliwe do znalezienia z rozsądnym wysiłkiem obliczeniowym,
- rozwiązanie powinno być bliskie optymalnemu,
- prawdopodobieństwo uzyskania złego rozwiązania powinno być niskie.

1. Opis problemu

Tu jest wnętrze rozdziału.

Twierdzenie 1. *Twierdzenie Twierdzenie Twierdzenie Twierdzenie Twierdzenie*

Definicja 1. *Twierdzenie Twierdzenie Twierdzenie Twierdzenie Twierdzenie*

1,1.0,1.0.0,

1.1. Tytuł podrozdziału

Francuzów sto wozów sieci purpurowe kwiaty każdy mimowolnie porządku pilnował

$$a^2 + b^2 = c^2. \tag{1}$$

Twierdzenie 1. *Twierdzenie Twierdzenie Twierdzenie Twierdzenie Twierdzenie*

Definicja 2. *Twierdzenie Twierdzenie Twierdzenie Twierdzenie Twierdzenie*

$$E = mc^2. \tag{2}$$

1.2. Cel

Przetestowanie algorytmu wyżarzania, bla bla pomimo, ze najlepiej sprawdza sie w zadaniach kombinatorycznych, to jak bedzie dzialac przy szukaniu minimum globalnym i zadaniach odwrotnych

2. Opis algorytmów

2.1. Algorytm symulowanego wyżarzania

Algorytm ten został stworzony wzorując się na zjawisku wyżarzania w metalurgii, które polega na nagrzaniu elementu stalowego do odpowiedniej temperatury, przetrzymaniu go w tej temperaturze przez pewien czas, a następnie powolnym jego schłodzeniu. Sam algorytm natomiast bazuje na metodach Monte-Carlo i w pewnym sensie może być rozważany jako algorytm iteracyjny.

Główną istotą i zarazem zaletą tego algorytmu jest wykonywanie pewnych losowych przeskoków do sąsiednich rozwiązań, dzięki czemu jest w stanie uniknąć wpadania w lokalne minimum. Algorytm ten najczęściej jest używany do rozwiązywania problemów kombinatorycznych, jak np. problemu komiwojażera.

2.1.1. Parametry

Początkowa konfiguracja

W tym kroku powinniśmy zainicjalizować naszą temperaturę wysoką wartością oraz znaleźć początkowe losowe rozwiązanie naszego problemu.

Temperatura

Temperatura jest zarówno czynnikiem iteracyjnym, jak i jest związana z funkcją prawdopodobieństwa zamiany gorszego rozwiązania na lepsze. Zatem zakres temperatury powinien być taki, aby na początku działania naszego algorytmu dawał wysoką możliwość zamian, a wraz z postępem iteracji te prawdopodobieństwo się zmniejszało i pod koniec było bliskie zeru.

Końcowa temperatura

Jest to bardzo mała wartość. Temperatura osiągając taki poziom informuje nas iż proces wyżarzania się zakończył i rozwiązanie zostało znalezione. Wartość ta powinna być na tyle mała, by temperatura będąc niewiele większa prowadziła do bardzo niskiego prawdopodobieństwa, a jednocześnie nie wymagało to zbyt dużej ilości iteracji.

Powtarzanie pewną ilość razy dla zadanej temperatury

Wartość ta powinna być z góry ustalona i powinna dać nam możliwość sprawdzenia wielu sąsiadów obecnego rozwiązania, równocześnie nie powodując zbyt dużego obciążenia dla algorytmu.

Znajdowanie losowego sąsiada poprzedniego rozwiązania

Funkcja ta powinna nam pozwalać przejrzeć jak najszerszy zakres rozwiązań, a jednocześnie pozwolić na przeszukiwanie coraz to bliższych sąsiadów obecnie najlepszego rozwiązania, zatem warto uzależnić tą funkcję od stopnia ukończenia globalnych iteracji.

Funkcja kosztu

Poprzez funkcję kosztu rozumiemy różnicę pomiędzy obecnie najlepszym rozwiązaniem, a nowym. Przy poszukiwaniu globalnego minimum wartość większa jest gorszym rozwiązaniem, dzięki czemu wynikiem tej funkcji jest zawsze liczba ujemna. W tym wypadku nowe rozwiązanie zawsze jest gorsze (a tym samym przy poszukiwaniu minimum globalnego posiada wartość większą), to w tym wypadku wartość jest ujemna.

Prawdopodobieństwo zamiany P

Prawdopodobieństwo jest potrzebne do decyzji czy zamienić nasze nowe i gorsze rozwiązanie z wcześniejszym, lepszym.

Prawdopodobieństwo zależy od funkcji kosztu oraz obecnej temperatury. Prawdopodobieństwo zatem można przedstawić w następujący sposób:

$$P = e^{\frac{\Delta E}{T}}$$

gdzie:

$$\Delta E - funkcja\ kosztu \tag{3}$$

T - obecna wartość temperatury

Prawdopodobieństwo to wraz ze spadkiem wartości funkcji kosztu maleje (gdyż jest zawsze ujemne), natomiast wyższa wartość temperatury zwiększa prawdopodobieństwo. Decydując o tym czy powinniśmy zamienić nasze gorsze rozwiązanie z lepszym powinniśmy porównać obliczone prawdopodobieństwo z wartością losową zawierającą się w zakresie [0,1].

Zmniejszanie temperatury

Szybkość zmniejszania nie powinna być zbyt duża, aby pozwolić algorytmowi na sprawdzenie jak największego zakresu możliwych rozwiązań, a jednocześnie niezbyt wolna, gdyż może to spowodować zbyt wolny spadek prawdopodobieństwa, a tym samym finalnie pozwolić na pozostanie w gorszym rozwiązaniu. W większości opracowań można spotkać ten proces, jako mnożnik temperatury w zakresie [0.8;0.99].

2.1.2. Kroki algorytmu

Algorytm ten można również przedstawić za pomocą listy kroków:

1. Zainicjalizuj początkową konfigurację
2. Dopóki temperatura ¿ minimum, powtarzaj:
   (a) Powtórz pewną ilość razy dla danej temperatury
      i. Znajdź losowo sąsiada poprzedniego rozwiązania
      ii. Sprawdź czy rozwiązanie jest lepsze od poprzedniego (funkcja kosztu)
         A. Jeżeli jest, zamień rozwiązania
         B. Jeżeli nie jest, zamień rozwiązania z pewnym prawdopodobieństwem P
   (b) Zmniejsz temperaturę

3. Funkcje testowe

3.1. Funkcja kwadratowa dwóch parametrów

Funkcja ta ma następującą postać:

$$f(x, y) = x^2 + y^2$$

Cośo wykresie: ¡tutaj jakis wykres ¿

Funkcja ta przyjmuje jedynie wartości dodatnie, zakres dla potrzeb projektu został zawężony do [-10, 10], natomiast minimum globalne posiada w punkcie [0, 0] i wynosi 0.

3.2. Funkcja Rastrigina

Funkcja Rastrigina jest funkcją ciągłą, skalowalną i multimodalną. Dzięki posiadaniu wielu minimum lokalnych funkcja ta jest często stosowana w testowaniu algorytmów optymalizacyjnych.

$$f(x) = An + \sum_{i=1}^{n} [x_i^2 - A\cos(2\pi x_i)]$$

gdzie:
A = 10,
n = ilość wymiarów

Parametry funkcji możemy znaleźć w zakresie [-5.12; 5.12], minimum globalne natomiast w punkcie [0,...,0] i wynosi 0.

3.3. Funkcja Rosenbrocka

Rosenbrock's valley function is known as the second function of De Jong. This test function is continuous, scalable, naturally nonseparable, nonconvex, and unimodal.

$$f(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$$

4. Dobór parametrów

5. Implementacja

6. Dostosowanie algorytmów do funkcji testowej zadań odwrotnych

7. Narzędzia i technologie

7.1. Użyte narzędzia

Aby zapewnić bezpieczeństwo oraz możliwość pracy w kilku miejscach nad jednym problemem, zastosowałem kilka rozwiązań.

7.1.1. System kontroli wersji

Git

7.1.2. Podział pracy na mniejsze zadania

Git

7.2. Użyte technologie

.NET Framework

8. Podsumowanie

8.1. Dalsze kierunki rozwoju

8.2. Źródła

https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4538776/

F. Rothlauf, Design of Modern Heuristics, Natural Computing Series, DOI 10.1007/978-3-540-72962-4 2, © Springer-Verlag Berlin Heidelberg 2011

R. Mart'ı and G. Reinelt, The Linear Ordering Problem, Exact and Heuristic Methods in Combinatorial Optimization 175, DOI: 10.1007/978-3-642-16729-4 2, c Springer-Verlag Berlin Heidelberg 2011

9. Tytuł drugiego rozdziału. Bardzo długi tytuł. Jego formatowanie jest trudniejsze

Dodatek

Mój specjalny dodatek

Tu treść dodatku. Zwróćmy uwagę na sposób numerowania dodatku, możliwa jest zmiana numerowania, patrz wyjaśnienia.

Rysunki

Tu rysunki

Programy

Tu programy

```
#include <stdio.h>

int main()
{
   printf("Hello world\n");
}
```

Oraz

```
<?php
␣␣␣echo␣"test=$test";
?>
```

Twierdzenie 2. *Twierdzenie Twierdzenie Twierdzenie Twierdzenie Twierdzenie*

Literatura

[1] Jakaś pozycja literatury

[2] Jakaś pozycja literatury