Algorytmy, przestrzenie przeszukiwania i metody jednopunktowe

Michał Okulewicz

Wydział Matematyki i Nauk Informacyjnych
Politechnika Warszawska

Plan wykładu

1. Algorytmy optymalizacyjne
   - Problemy optymalizacyjne
   - Klasyfikacja algorytmów
2. Przestrzenie i operatory przeszukiwania
   - Przestrzenie ciągłe
   - Przestrzenie dyskretne i kombinatoryczne
3. Metody jednopunktowe
   - Monte Carlo
   - Losowe błądzenie
   - Hill climbing
   - Symulowane wyżarzanie
   - Variable Neighbourhood Search

Definicja (Problem optymalizacyjny)

Problem optymalizacyjny $\mathcal{P} = (\Omega, f, \succ)$ jest zdefiniowany przez przestrzeń rozwiązań Ω, funkcję jakości[a] $f : \Omega \to \mathbb{R}$ oraz relację $\succ \in \{<, >\}$. Rozwiązaniem problemu jest znalezienie zbioru elementów (elementu) $\mathcal{X} \subseteq \Omega$ spełniającego następujące warunki:

$$\mathcal{X} = \{x \in \Omega : \forall_{x' \in \Omega} f(x) \succeq f(x')\}$$

[a] W zależności od dziedziny funkcja jakości może być też określana jako funkcja celu lub funkcja przystosowania.

Definicja (Podzbiór dopuszczalny)

Podzbiorem dopuszczalnym zbioru Ω jest zbiór $\mathcal{D}$ spełniający następujące warunki:

$$\mathcal{D} = \{x \in \Omega : f(x) \text{ ma sens matematyczny i/lub biznesowy}\}$$

Algorytmy optymalizacyjne I

Poszukiwanie rozwiązania:

- Konstrukcyjne
- Iteracyjnego poprawiania

Gwarancja jakości rozwiązania:

- Heurystyczne
- Aproksymacyjne
- Probabilistyczne
- Dokładne

Algorytmy optymalizacyjne II

Zakres przeszukiwania zbioru rozwiązań:

- Lokalne
- Globalne

Sposób wykorzystania funkcji jakości:

- Gradientowe
- Próbkujące

Wiedza o funkcji jakości:

- Jednopunktowe
- Populacyjne

Definicja (Metaheurystyki)

Za podręcznikiem i wykładami prof. Arabasa [1, 2], **metaheurystyka** będzie rozumiana jako pewien ramowy (niezależny od rozwiązywanego problemu) sposób przeszukiwania przestrzeni rozwiązań. W sposobie tym należy określić (zależne już od problemu) kodowanie rozwiązania, operatory przeszukiwania tej przestrzeni oraz rozmiar pamięci algorytmu.

Przestrzenie przeszukiwania I

Przestrzenie przeszukiwania II

Definicja

Przestrzenią przeszukiwań $\mathcal{G}$ problemu $\mathcal{P}$ jest zbiór, na którym zdefiniowana jest operacja $dec : \mathcal{G} \to \Omega$ transformująca element przestrzeni przeszukiwań $g \in \mathcal{G}$ w element przestrzeni rozwiązań $x \in \Omega$.

Przestrzenie przeszukiwania III

Definicja

Algorytm optymalizacyjny $\mathcal{A}$ to zbiór operatorów przeszukiwania $\mathcal{O}$ pracujących na elementach przestrzeni przeszukiwań $\mathcal{G}$ oraz wykorzystujący złożenie $f \circ dec : \mathcal{G} \to \mathbb{R}$ do ewaluacji elementów tej przestrzeni i zwracający zbiór $\mathcal{X}^*$, będący aproksymacją poszukiwanego zbioru $\mathcal{X}$ o następujących cechach:

$$\mathcal{X}^* = \{x \in \Omega_A : \forall_{x' \in \Omega_A} f(x) \succeq f(x')\}$$

Ω_A jest podzbiorem przestrzeni rozwiązań przeliczonych przez algorytm $\mathcal{A}$ w trakcie jego działania. Operator $O \in \mathcal{O}$ jest postaci $O : \mathcal{P} \times \mathcal{G}^{n_0} \to \mathcal{G}^{n_1}$, gdzie $n_0 \in \mathbb{N}_0$, $n_1 \in \mathbb{N}_+$.

Przykład

Przykładem operatora (ozn. I) korzystającego wyłącznie z danych problemu i generującego element przestrzeni przeszukiwań $I : P \to \mathcal{G}$ są wszelkiego rodzaju inicjalizacje rozwiązań oraz algorytmy jednokrokowe.

Przykład

Przykładem operatora (ozn. M) korzystającego z pojedynczego elementu przestrzeni przeszukiwań i zwracającego jeden taki element $M : \mathcal{G} \to \mathcal{G}$ jest mutacja w algorytmie genetycznym.

Przykład

Przykładami operatorów wykorzystujących wiele elementów przestrzeni przeszukiwań i zwracających jeden lub wiele elementów jest krzyżowanie wymieniające (ozn. C) w algorytmie genetycznym $C : \mathcal{G} \times \mathcal{G} \to \mathcal{G} \times \mathcal{G}$, czy zmiana prędkości (ozn. Δ_V) w optymalizacji rojem cząstek $\Delta_V : \mathcal{G} \times \mathcal{G} \times \mathcal{G} \to \mathcal{G}$.

Przestrzenie ciągłe

Definicja

Problem ciągły charakteryzuje się tym, że jego rozwiązanie jest reprezentowane w formie wektora liczb rzeczywistych ($\Omega = \mathbb{R}^n$).

Przestrzenie ciągłe

Zbiór rozwiązań:

- $\mathbb{R}^n$

Typowe zastosowania:

- Procesy sterowania

Typowe operatory:

- Gaussowskie zaburzanie
- Uśrednianie rozwiązań

Przestrzenie dyskretne i kombinatoryczne

Definicja

Problem dyskretny charakteryzuje się tym, że jego rozwiązanie jest reprezentowane w formie wektora liczb całkowitych ($\Omega = \mathbb{Z}^n$).

Definicja

Problem kombinatoryczny charakteryzuje się tym, że jego rozwiązanie jest reprezentowane w formie wektora binarnego ($\Omega = \mathbb{Z}_2^n$).

Przestrzenie dyskretne i kombinatoryczne

Zbiór rozwiązań:

- $\{0,1\}^n$
  - Grafy (w tym np. sieci neuronowe)
- $\mathbb{Z}^n$
  - Słowa

Typowe zastosowania:

- Problemy logistyczne, produkcyjne i transportowe

Typowe operatory:

- Przełączanie bitów
- Permutacje
- Wymiany

Metody jednopunktowe

- Wykorzystują lokalne operatory
- Mogą być uruchamiane równolegle

Monte Carlo

- Najprostsza metoda w implementacji
- Niezależne losowanie kolejnych rozwiązań z przestrzeni problemu / przestrzeni przeszukiwania
- Konieczne założenia dotyczące lokalizacji zbioru dopuszczalnego

Losowe błądzenie

- Pozornie bezsensowna modyfikacja metody Monte Carlo
- Losowanie kolejnych rozwiązań w oparciu o poprzednio testowane rozwiązanie
- Jeżeli znamy przybliżony rozkład wartości funkcji możemy poprawić jakość działania algorytmu
- WPP równoważny metodzie Monte Carlo
- Brak założenia o znajomości zbioru dopuszczalnego

Hill climbing

- Naturalne rozszerzenie metody losowego błądzenia
- Losowanie kolejnych rozwiązań w oparciu o poprzednio testowane rozwiązanie, które poprawiło wynik algorytmu
- Algorytm lokalnego przeszukiwania

Simulated Annealing

- Połączenie losowego błądzenia z *hill climbing*
- Nowe rozwiązania są akceptowane zawsze, jeżeli poprawiają poprzedni wynik
- Nowe rozwiązania są akceptowane z pewnym prawdopodobieństwem, nawet jeżeli pogarszają poprzedni wynik
- Stopniowe zawężanie zakresu działania algorytmu

1: $T \leftarrow T_{init}$ ▷ Początkowa temperatura zależy od względnych wartości funkcji
2: $x_{best} \leftarrow x \leftarrow x_{init}$ ▷ Wybór punktu startowego
3: **while** $T > T_{min}$ **do**
4: **for** $iter \in \{1, 2, \ldots, iter_{max}\}$ **do** ▷ Przez jakiś czas działamy w tej samej temperaturze
5: $x' \leftarrow Sample(x, T)$ ▷ Nowa próbka zależy od aktualnej oraz od temperatury
6: **if** $\exp(-\frac{(f(x') - f(x))}{T}) > rand(0, 1)$ **then**
7: $x \leftarrow x'$ ▷ Akceptujemy każdą poprawę oraz pogorszenie adekwatne do temperatury
8: **end if**
9: **if** $f(x) < f(x_{best})$ **then**
10: $x_{best} \leftarrow x$ ▷ Zapamiętujemy najlepszy punkt
11: **end if**
12: **end for**
13: $x \leftarrow x_{best}$ ▷ Kolejną pętlę rozpoczynamy od najlepszego dotychczasowego punktu
14: $T \leftarrow cool * T$ ▷ Parametr *cool* powinien być mniejszy od 1, ale niewiele
15: **end while**

Variable Neighbourhood Search

- Alternatywny sposób przełamania lokalności algorytmu *hill climbing*
- Poszerzanie zakresu działania operatora modyfikującego rozwiązanie

1: $x_{best} \leftarrow x \leftarrow x_{init}$ ▷ Wybór punktu startowego
2: **while** STOP criteria not met **do**
3: **for** $neighbour \in \{1, 2, \ldots, neighbours_{max}\}$ **do**
4: $improvement \leftarrow$ **false**
5: $x' \leftarrow Sample(x, Neighbourhood)$ ▷ Próbka z sąsiedztwa
6: **if** $f(x') < f(x_{best})$ **then**
7: $x_{best} \leftarrow x'$ ▷ Szukamy największej poprawy
8: $improvement \leftarrow$ **true**
9: **end if**
10: **end for**
11: $x \leftarrow x_{best}$
12: **if** no *improvement* **then**
13: $Neighbourhood \leftarrow NextNeighbourhood$ ▷ Zmieniamy sąsiedztwo
14: **else**
15: $Neighbourhood \leftarrow InitialNeighbourhood$ ▷ Wracamy do bazowego sąsiedztwa
16: **end if**
17: **end while**

Praca domowa

- Porównać działanie Symulowanego Wyżarzania z *Variable Neighbourhood Search* oraz swoim najlepszym *Hill Climbingiem*
- Należy poeksperymentować z doborem parametrów obu metod

Bibliografia I


Jarosław Arabas.
Wykłady z algorytmów ewolucyjnych.
Wydawnictwa Naukowo-Techniczne, 2004.


Jarosław Arabas.
Metaheuristics – a modern approach.
jun 2014.


Lester Ingber.
Adaptive simulated annealing (asa): Lessons learned.
arXiv preprint cs/0001018, 2000.

Bibliografia II


S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi.
Optimization by simulated annealing.
Science, 220(4598):671–680, 1983.


Nenad Mladenović and Pierre Hansen.
Variable neighborhood search.
Computers & operations research, 24(11):1097–1100, 1997.


Trung Thanh Nguyen, Shengxiang Yang, and Jürgen Branke.
Evolutionary dynamic optimization: A survey of the state of the art.
Swarm and Evolutionary Computation, 6:1–24, 08 2012.


Karsten Weicker.
Evolutionary Algorithms and Dynamic Optimization Problems.
PhD thesis, Universitat Stuttgart, 2003.