[ALHE] Dylemat ciasteczkowy

Bartosz Świtalski Marcel Kawski

March 2021

1 Wprowadzenie

Rozważamy następujący problem:

Nauczycielka w przedszkolu musi rozdać dzieciom ciastka wg uzyskanych przez nie wyników z testu umiejętności. Dzieci siedzą w linii obok siebie (i nie zmieniają tych pozycji). Zgodnie z przyjętymi zasadami, jeśli dwoje dzieci siedzi obok siebie, dziecko z wyższą oceną musi dostać więcej ciastek. Nauczycielka ma ograniczony budżet i chce rozdać jak najmniej ciastek.

W poniższej pracy, bazując na dwóch wybranych rodzajach algorytmów ewolucyjnych, zaimplementujemy program generujący rozwiązanie takiego problemu, a następnie porównamy te metody do rozwiązania optymalnego, wygenerowanego metodą `MIP`.

W celu określenia funkcji oceny (ewaluacji) rozwiązania (genotypu) zaproponujemy własną heurystykę.

2 Decyzje projektowe

- Wybrany język programowania to `Python`
- Wybrane rodzaje algorytmów ewolucyjnych do implementacji to:
  1. strategia ewolucyjna $(\mu + \lambda)$
  2. klasyczny algorytm genetyczny (Holland, 1975)
- Przyjęty budżet możliwych ewaluacji funkcji celu dla pojedynczej próby optymalizacji wynosi $1000 * D$(wymiarowość zadania)
- Będziemy testować różne wymiarowości (zróżnicowane rzędem wielkości)
- Podczas jednego uruchomienia programu będziemy dokonywać uśrednienia wyników z 25 wywołań algorytmu
- Zaimplementujemy kryterium stopu `k-iteracji`
- W przypadku zajścia kryterium stopu obliczenia w danych kroku rozpoczynają się od nowa (o ile pozwala na to pozostały budżet)
- Ocena możliwa do uzyskania przez i-tego ucznia z testu umiejętności to $g_i \in \{1, 2, 3...10\}$
- Ciastko jest niepodzielną jednostką (nie można dać uczniowi $1\frac{2}{3}$ ciastka)
- Naszym celem jest minimalizacja funkcji celu q

2.1 Generowanie rozwiązań optymalnych metodą MIP

- Cel: minimalizacja sumy przydzielonych ciastek wszystkim uczniom:

  $min(\sum_i c_i)$

  , gdzie c_i to liczba ciastek przydzielona i-temu uczniowi.
- Ograniczenia:
  - Dziedzinowe: $1 \leq c_i \leq 10$
  - Liniowe: Iterujemy po zbiorze ocen $G = g_1, g_2, ..., g_i$ uzyskanych przez uczniów, rozpoczynając od oceny drugiego ucznia (g_2). W każdym kroku iteracji warunkowo dodajemy następujące ograniczenia liniowe:
    - jeśli $g_i > g_{i-1}$, to dodajemy ograniczenie $c_i - c_{i-1} >= 1$
    - jeśli $g_i < g_{i-1}$, to dodajemy ograniczenie $c_i - c_{i-1} <= -1$

2.2 chromosom osobnika

Dla obydwu implementowanych algorytmów ewolucyjnych chromosom osobnika jest listą liczb całkowitych z zakresu $[1;10]$ o długości równej wymiarowości zadania. Jest on reprezentacją proponowanego przydziału ciastek.

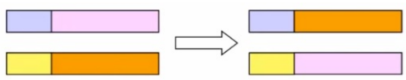
3 Zastosowane rodzaje algorytmów ewolucyjnych

3.1 strategia ewolucyjna $(\mu + \lambda)$

- Strategia elitarna
- Populacja bazowa ma μ osobników, a potomna λ osobników
- Osobnik zawiera 2 chromosomy. Ten dodatkowy zawiera wartości σ używane do mutacji
- Krzyżowanie uśredniające z losową wagą: $y = w \cdot x_1 + (1 - w) \cdot x_2$, gdzie w jest losowane z rozkładu jednostajnego $U(0, 1)$
- Mutacja ma 3 etapy. Dla każdego osobnika:
  1. $a = N(0, 1); b_j = N(0, 1)_j, j \in 1...J$, gdzie J to liczba cech osobnika
  2. $\sigma_j \leftarrow \sigma_j \exp(\tau' a + \tau b_j)$, gdzie $\tau = \frac{1}{\sqrt{2n}}$, a $\tau' = \frac{1}{\sqrt{2\sqrt{n}}}$
  3. $O_j = T_j + \sigma_j N(0, 1)_j$

3.2 algorytm genetyczny (Holland, 1975)

- Selekcja ruletkowa - prawdopodobieństwo wyboru osobnika jest wprost proporcjonalne do wartości funkcji przystosowania: $p_s(P(t, i)) = \frac{q(P(t,i))}{\sum_j q(P(t,j))}$ (wzór dotyczy maksymalizacji, przy minimalizacji należy go przekształcić)
- Krzyżowanie jednopunktowe - wybieramy losowo punkt przecięcia genotypu, z dwóch osobników rodzicielskich powstają dwa osobniki potomne przez prostą wymianę części genotypów rodziców



- Sukcesja generacyjna
- Duży rozmiar populacji
- Duże prawdopodobieństwo krzyżowania, bardzo małe prawdopodobieństwo mutacji
- Mutacja Gaussowska: jeśli dla danego osobnika zachodzi mutacja, to do każdej z wartości w chromosomie jest dodawana losowa wartość z rozkładu normalnego

4 Heurystyka funkcji oceny

W celu oceny osobnika zastosujemy następującą heurystykę:

- Jeżeli rozwiązanie jest niedopuszczalne, to funkcja kara w dużym stopniu tak, że bardziej opłaca się przydzielić więcej ciastek, aniżeli nagiąć choć trochę zasady:
  ocena $= \sum_{j=1}^{D-1} 10\cdot$(różnica ciastek w niedozwolonym przydziale między uczniami u_j i u_{j+1} $+ 1) + \sum_{j=1}^{D}$(liczba przydzielonych ciastek dla j-tego ucznia u_j)
- Jeżeli rozwiązanie jest dopuszczalne, to im mniejsza suma przydzielonych ciastek, tym lepsza ocena (mniejsza wartość funkcji celu q):
  ocena $= \sum_{j=1}^{D}$(liczba przydzielonych ciastek dla j-tego ucznia)

5 Testowanie

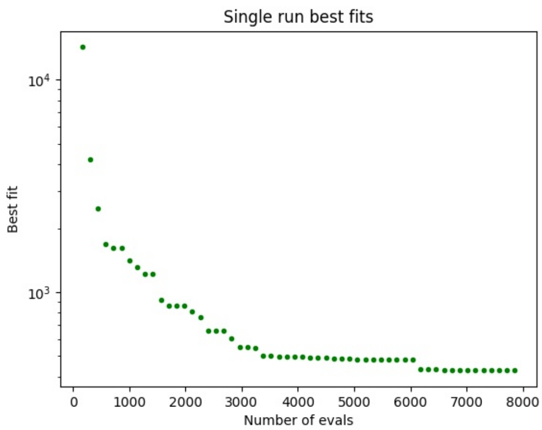
Testowaliśmy na wybranych ustawieniach początkowych:

1. strategia ewolucyjna $(\mu + \lambda)$:
   - $\mu = 20$
   - $\lambda = 7 \cdot \mu$
   - $\sigma_0 \in (0.9; 1.1)$
2. algorytm genetyczny
   - rozmiar populacji = 10∗wymiarowość
   - prawdopodobieństwo krzyżowania = 0.9
   - prawdopodobieństwo mutacji = 0.01

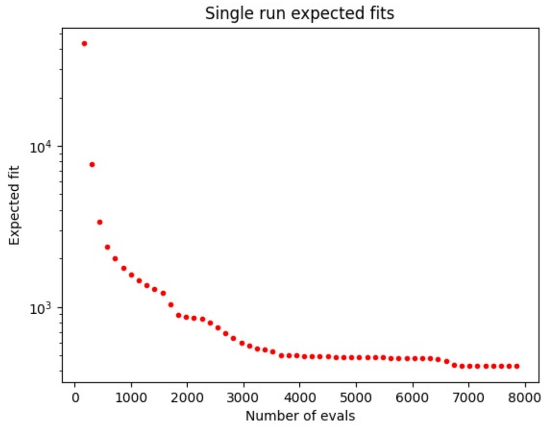
6 Wyniki

Testowaliśmy następujące wymiarowości zadania: $\{5, 10, 20, 40, 80, 160\}$.

6.1 Przykłady działania strategii ewolucyjnej $(\mu + \lambda)$

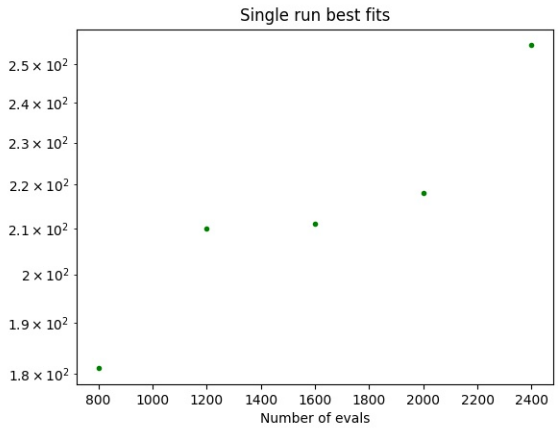


Rysunek 1: Dopasowanie najlepszego osobnika populacji w zależności od liczby ewaluacji funkcji celu dla wymiarowości $D = 50$.

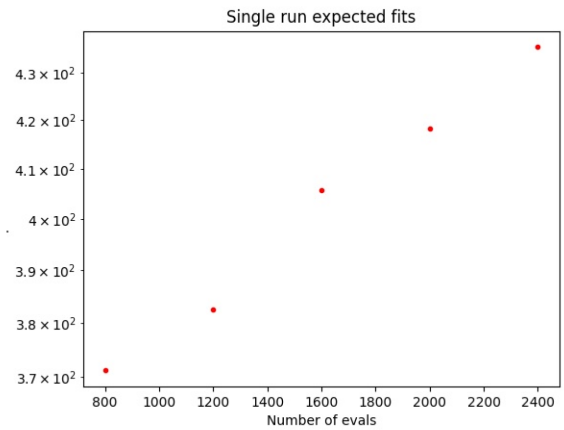


Rysunek 2: Dopasowanie średniego osobnika populacji w zależności od liczby ewaluacji funkcji celu dla wymiarowości $D = 50$.

6.2 Przykłady działania algorytmu genetycznego



Rysunek 3: Dopasowanie najlepszego osobnika populacji w zależności od liczby ewaluacji funkcji celu dla wymiarowości $D = 40$.



Rysunek 4: Dopasowanie średniego osobnika populacji w zależności od liczby ewaluacji funkcji celu dla wymiarowości $D = 40$.

6.3 Wyniki testowe

objaśnienia:

- `D` - wymiarowość zadania
- `best fit` - najlepsze znalezione rozwiązanie
- `best fit mean` - średnia najlepszych znalezionych rozwiązań
- `best fit std. deviation` - odchylenie standardowe najlepszych znalezionych rozwiązań

6.3.1 losowa inicjacja osobników z wartościami w chromosomie z zakresu [1;5], krzyżowanie dwupunktowe w algorytmie genetycznym

		strategia ewolucyjna $(\mu + \lambda)$			**algorytm genetyczny**		
`D`	optimum	best fit	best fit mean	best fit std. deviation	best fit	best fit mean	best fit std. deviation
5	7	7	7	0	7	7.84	0.54
10	15	15	15.16	0.37	19	22.6	2.06
20	34	37	40.56	2.35	67	82.04	8.99
40	70	91	102.96	6.02	198	234.64	16.24
80	139	261	303.48	20.45	547	665.04	34.12
160	239	864	980.36	46.57	1565	1657.4	44.35

7 Wnioski

Zauważyliśmy, że algorytm genetyczny zdecydowanie gorzej rozwiązuje nasz dylemat rozdawania ciastek. Dla wymiarowości > 10 zauważalne są różnice między obydwoma algorytmami. Jedną z przyczyn takiego stanu sytuacji może być fakt, że w naszej wersji algorytmu genetycznego przyjmujemy, że mutacja występuje bardzo sporadycznie ($\approx 1\%$), a głównym mechanizmem napędzającym ewolucję jest krzyżowanie (jednopunktowe, $\approx 90\%$ przypadków). W takim wypadku, jeśli wśród populacji nie ma takiego osobnika, który posiada w swoim genotypie optymalną dla danego problemu wartość jednej z cech (pojedyncza cecha=liczba rozdanych ciastek jednemu dziecku), szansa na uzyskanie optimum globalnego, czy też lokalnego, jest niska. Jeśli populacja bazowa zostanie wylosowana niekorzystnie (co jest bardzo prawdopodobne z uwagi na naturę problemu), to nawet przy dłuższym działaniu algorytmu (większej liczbie pokoleń) nie następuje poprawa.

Z drugiej strony mamy do czynienia ze strategią ewolucyjną, czyli z podejściem z sukcesją elitarną. Jak widać w wynikach strategia ta jest zdecydowanie bardziej skuteczna, a dla małych wymiarowości znajduje optimum globalne z bardzo wysokim prawdopodobieństwem. Jest to prawdopodobnie spowodowane dużym prawdopodobieństwem mutacji, która w różnych strategiach ewolucyjnych jest uznawana jako kluczowy element ewolucji (w przeciwieństwie do algorytmów genetycznych).

7.1 krzyżowanie wielopunktowe

Z uwagi na początkowo słabe działanie algorytmu genetycznego zdecydowaliśmy zwiększyć prawdopodobieństwo wystąpienia krzyżowania z 0.7 na 0.9 oraz przetestować krzyżowanie wielopunktowe (> 1). Testowaliśmy krzyżowanie dwupunktowe oraz trzypunktowe. Wyniki dla krzyżowania dwupunktowego zostały przedstawione w punkcie 6.3.1.

8 Podsumowanie

Projekt poruszający tematykę algorytmów heurystycznych. Dzięki własnej implementacji strategii ewolucyjnej i algorytmu genetycznego poznano istotę algorytmów ewolucyjnych. Zastosowanie autorskiej heurystyki funkcji oceny pozwoliło na dokonanie porównania pomiędzy tymi konkretnymi typami algorytmów ewolucyjnych.