AGNIESZKA BOROWIECKA jest nauczycielem konsultantem w Ośrodku Edukacji Informatycznej i Zastosowań Komputerów w Warszawie.

DR KATARZYNA OLĘDZKA jest nauczycielem konsultantem w Ośrodku Edukacji Informatycznej i Zastosowań Komputerów w Warszawie.

OD WYSZUKIWANIA DO SZTUCZNEJ INTELIGENCJI

AGNIESZKA BOROWIECKA ● KATARZYNA OLĘDZKA

Każdy z nas na co dzień spotyka się z problemem wyszukiwania, jednak niekoniecznie zdajemy sobie sprawę, że stoją za tym algorytmy. W pewnym sensie z wyszukiwaniem w zbiorze nieuporządkowanym mamy do czynienia przy różnego typu kodach i zabezpieczeniach, takich jak szyfrowe zamki w walizkach i blokadach rowerowych, kody pin do kart bankomatowych i telefonu, hasła do kont pocztowych. Bez znajomości kodu otworzenie zabezpieczenia sprowadza się do przeglądania wszystkich możliwości. Z wyszukiwaniem w zbiorze uporządkowanym mamy do czynienia przy różnego rodzaju szacowaniach, odgadywaniu liczby z podpowiedziami *za mało* lub *za dużo*, wyszukiwaniu haseł w tradycyjnej encyklopedii.

Gdy problemy stają się trudniejsze, komplikują się także algorytmy pozwalające je rozwiązać. Dlatego podejmowane są próby wykorzystania do tego celu sztucznej inteligencji. Przedstawimy kilka przykładów, które pozwalają stopniowo wprowadzać uczniów w te zagadnienia, pokazując, jak można analizować problemy i ułatwiać rozumienie złożonych zagadnień związanych ze sztuczną inteligencją.

JEDEN SPOŚRÓD DZIESIĘCIU

Zastanówmy się nad problemem odnalezienia konkretnego elementu wśród wielu do niego podobnych. Do pudełka wrzucamy 10 kulek tej samej wielkości, wśród których znajduje się jedna zielona. Potrząsamy pudełkiem, by kulki dokładnie się wymieszały, a następnie wybieramy ochotnika, którego zadaniem będzie wyciągnięcie zielonej kulki. Uczeń nie widzi, jakiego koloru kulkę wyciąga. Próbuje kolejno. Jeśli wyjęta kulka nie była właściwego koloru, kontynuuje wybieranie z pudełka. Powtarzamy eksperyment wielokrotnie i notujemy wyniki.

Próbujemy ustalić z uczniami odpowiedzi na następujące pytania: Ile prób należy wykonać, by wyciągnąć zieloną kulkę? Czy istnieje sposób losowania zmniejszający liczbę prób? Czy możliwe jest trafienie na zielony kolor przy pierwszym sięgnięciu do pudełka?

Prowadzimy rozmowę tak długo, aż nie padną następujące wnioski:

- Jeśli szukamy konkretnego elementu wśród wielu podobnych do niego, to możliwe jest znalezienie szukanego elementu w pierwszej próbie, ale jest to mało prawdopodobne.

AGNIESZKA BOROWIECKA
KATARZYNA OLĘDZKA

- W przypadku pesymistycznym, musimy wyjąć wszystkie elementy.

Modyfikujemy eksperyment, zwiększając liczbę zielonych kulek w pudełku. Tym razem zadanie polega na wyjęciu wszystkich zielonych kulek, a osoba wybierająca nie wie, jaka jest ich liczba. Jakie teraz nasuną się wnioski?

Łatwo zauważyć, że szukanie wśród 10 elementów jest szybkie. Jednak gdy jest ich więcej, problemem może być czasochłonność całego procesu. Warto zachęcić uczniów do wskazania przykładów z życia codziennego związanych z wyszukiwaniem w zbiorze nieuporządkowanym.

Algorytm wyszukiwania w zbiorze nieuporządkowanym możemy implementować w różnych językach programowania. Ze względu na reprezentację danych w komputerze najprościej jest przeglądać liczby (tablice lub listy) albo znaki (napisy). Tworzony program zależy od tego, czy interesuje nas odnalezienie pierwszej prawidłowej wartości, czy wyszukanie wszystkich możliwych wystąpień. Przykładowa funkcja **szukaj()** zapisana w języku Python wypisuje pozycje wszystkich wystąpień litery „a" w podanym słowie.

```
1.  def szukaj(wyraz):
2.    for i in range(len(wyraz)):
3.     if wyraz[i] == 'a':
4.      print(i + 1)
```

Niewielka modyfikacja powyższego kodu pozwoli zapamiętać odnalezione wartości na liście:

```
1.  def szukaj(wyraz):
2.    pozycje = []
3.    for i in range(len(wyraz)):
4.     if wyraz[i] == 'a':
5.      pozycje.append(i + 1)
6.    return pozycje
```

lub podać liczbę wystąpień szukanej litery:

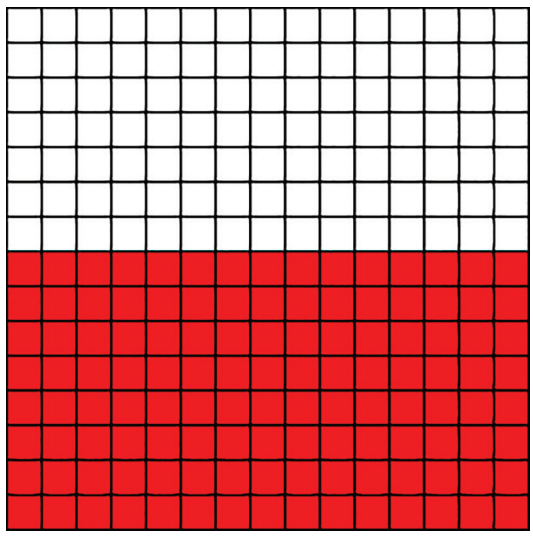
```
1.  def szukaj(wyraz):
2.    ile = 0
3.    for i in range(len(wyraz)):
4.     if wyraz[i] == 'a':
5.      ile = ile + 1
6.    return ile
```

Zapisując algorytmy w języku formalnym, a następnie je implementując, kształtujemy myślenie komputacyjne. Uczymy analizy algorytmów, dzielenia na podproblemy i szukania optymalnego rozwiązywania. Pomagamy uczniom lepiej zrozumieć świat, pokazując, jak za pomocą metod wywodzących się z informatyki rozwiązywać różne problemy.

WYSTĄP Z SZEREGU

Wprowadzenie uporządkowania w zbiorze pozwala nam zmienić sposób wyszukiwania. Przejrzenie wszystkich elementów przestaje być niezbędne do osiągnięcia celu, jednak konieczne jest opracowanie strategii postępowania i jej dobra implementacja.

Przypuśćmy, że mamy narysować kwadratowy wycinek polskiej flagi. Został on opisany za pomocą szablonu, który zawiera litery „b" – biały oraz „c" – czerwony, odpowiadające barwom wszystkich punktów na lewej skrajnej krawędzi kwadratu. Kolory są zapamiętywane w kolejności od góry do dołu.

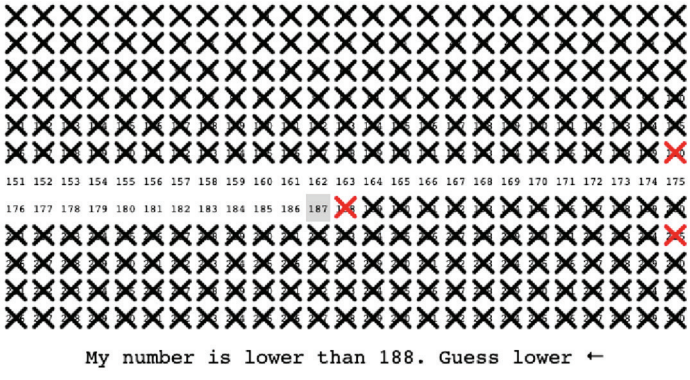


RYSUNEK 1. Przykładowy wycinek flagi opisany szablonem *bbbbbbbccccccc*

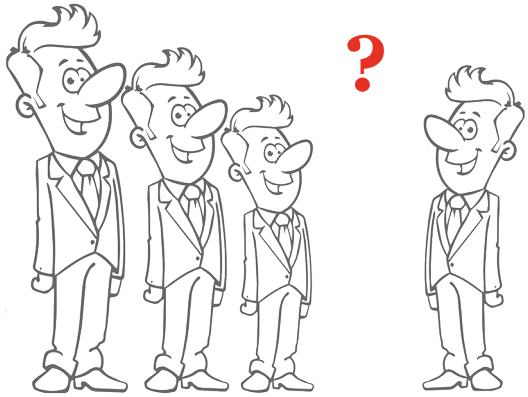
OD WYSZUKIWANIA DO SZTUCZNEJ INTELIGENCJI

Naszym zadaniem jest ustalenie miejsca, w którym zaczyna się czerwony kolor, co pozwoli policzyć ilość farby niezbędnej do namalowania fragmentu flagi. Narzucającym się w pierwszej chwili sposobem postępowania będzie przeglądanie po kolei wszystkich literek, aż nie napotkamy „c". Jednak łatwo zauważyć, że czasem szybciej znajdziemy rozwiązanie, zaczynając wyszukiwanie od początku, a czasem od końca szablonu. Wszystko zależy od tego, jaki fragment flagi został zapamiętany. Powstaje pytanie, czy można szybciej? Okazuje się, że można zastosować algorytm wyszukiwanie binarnego – z niesprawdzonych elementów wybieramy środkowy, jeśli jest on czerwony postępowanie kontynuujemy dla lewej połówki, gdy biały – prawej. Kończymy, gdy długość przedziału wynosi dwa – lewy element jest biały, a prawy – czerwony.

Podobnym zagadnieniem jest odgadywanie liczby z podanego zakresu. Uczniowie mogą spróbować kilkukrotnie zgadywać, za każdym razem notując liczbę wykonanych prób oraz przebieg gry (kolejne podawane przez osobę odgadującą wartości). Porównując zapisy poszczególnych rozgrywek, możemy spróbować wypracować najbardziej optymalną strategię postępowania. Warto przy tym skorzystać z komputerowej wizualizacji, demonstrującej, w jaki sposób po każdej próbie zmniejsza się zasób dostępnych jeszcze wartości. Przykład takiej aplikacji znajdziemy w kursie **Informatyka** na portalu Akademia Khana, w części **Gra w zgadywanie liczb**.

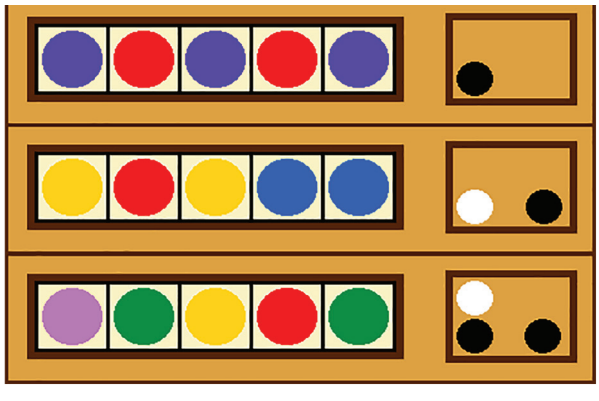


A jak jest z zagadnieniem odwrotnym – tworzeniem uporządkowanego zbioru z użyciem metody sortowania przez wstawianie? W metodzie tej możemy zastosować wyszukiwanie binarne do znalezienia pozycji wstawianego elementu. Uczniowie mogą wypróbować działanie tego algorytmu, ustawiając się w rzędzie według wzrostu. Kolejny uczeń dołączający do szeregu musi podjąć decyzję, w którym miejscu powinien stanąć.



GDY SZUKANIE STAJE SIĘ TRUDNIEJSZE

O ile przy zgadywaniu liczby po kilku grach strategia staje się coraz bardziej intuicyjna, tak w grze Mastermind nie jest to już takie proste. Rozważamy kombinacje kolorowych kulek. Jeden z graczy układa kod, a drugi próbuje go zgadnąć. W pojedynczym ruchu osoba odgadująca ustawia układ kulek i w ten sposób zadaje pytanie. Jeśli dana kulka jest na właściwym miejscu, to otrzymujemy za nią czarny pionek, jeśli kolor się zgadza, ale położenie nie – biały. Pełna odpowiedź jest sumą czarnych i białych pionków.



AGNIESZKA BOROWIECKA
KATARZYNA OLĘDZKA

Przyjrzyjmy się implementacji Artifico Mastermind game[1]. Zamiast kolorowych kulek mamy do dyspozycji sześć różnych ikonek, w odpowiedzi czarne i białe pionki zastąpiono czerwonymi i żółtymi kółkami. Wybieramy jeden z trzech trybów gry: sztuczna inteligencja może grać z komputerem (AI vs Computer), sztuczna inteligencja może grać z człowiekiem (AI vs Human) lub człowiek z komputerem (Human vs Computer). W kolejnych ruchach „gracz" układa kod, a druga strona odpowiada mu komunikatem – liczbą pionków czerwonych (dobry kod na dobrym miejscu) oraz żółtych (dobry kod na złym miejscu).

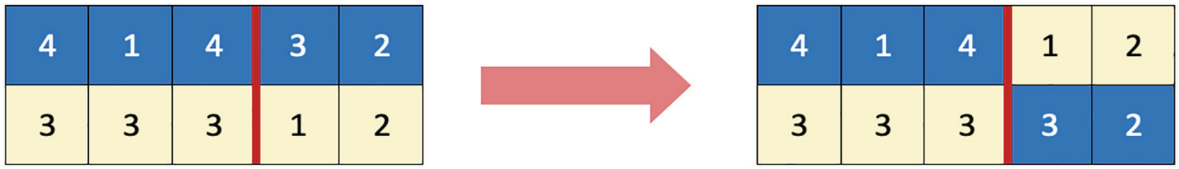
Aplikacja stwarza okazję do wielu eksperymentów, np. za którym razem jesteśmy w stanie odgadnąć kod, a za którym razem przeciętnie kod odgaduje AI. Jaka jest optymalna strategia dla nas? Czy możemy opisać strategię stosowaną przez AI?

Kod źródłowy aplikacji został udostępniony wszystkim zainteresowanym, umieszczono go na githubie. Możemy pobrać go na swój komputer i np. zmodyfikować liczbę kodów do zgadnięcia.

Interesuje nas pytanie, jak działa sztuczna inteligencja. Okazuje się, że w prezentowanej aplikacji został zaimplementowany algorytm genetyczny. Nie jest to rozwiązanie deterministyczne, lecz opierające się na rachunku prawdopodobieństwa. Nazwa algorytmu wskazuje na jego podobieństwo do procesów ewolucyjnych, które zachodzą w przyrodzie.

W algorytmie genetycznym mamy do czynienia z krzyżowaniem, mutacją i permutacją chromosomów. W przypadku gry Mastermind poprzez chromosom będziemy rozumieć pojedynczy kod. Krzyżowanie to proces, w którym dla dwóch chromosomów wybieramy punkt krzyżowania, a następnie zamieniamy miejscami fragmenty kodu.

Schemat algorytmu dla gry Mastermind wygląda następująco:



RYSUNEK 2. Przykład krzyżowania dla 5-elementowej gry

RYSUNEK 3. Przykład mutacji i permutacji dla 5-elementowej gry

[1] http://ognjenvucko.github.io/mastermind-ai

Mutacja to operacja, w której jeden gen jest losowo zastępowany przez inny, a permutacja polega na zamianie kolejności genów.

1. Wygeneruj populację złożoną z różnych możliwych wartości kodów – chromosomów.
2. Wybierz z nich jeden i sprawdź jego ocenę – liczbę czarnych i białych pionków. Jeśli liczba czarnych jest równa długości kodu, to kod został znaleziony.
3. Promując osobniki lepiej przystosowane, za pomocą krzyżowania, mutacji i permutacji zmodyfikuj populację.
4. Powróć do punktu 2.

Okazuje się w praktyce, że tak zaprojektowany algorytm genetyczny daje dobre rezultaty. Zresztą proszę spróbować samodzielnie.

PODSUMOWANIE

Otaczający nas świat zmienia się błyskawicznie, nikogo już nie dziwią reklamy dopasowane do naszych wcześniejszych zapytań, podpowiedzi Google, jak znaleźć drogę do domu czy komputer wygrywający w różnego rodzaju gry z człowiekiem. Niedługo w naszych domach zagości lodówka sama zamawiająca zakupy, na ulicach – samochód bez kierowcy, a asystent Google będzie nie tylko atrakcyjny, ale też użyteczny. Apokaliptyczne wizje ukazywane od dawna w filmach, w których „inteligentne" maszyny dominują nad człowiekiem, wydają się coraz bliższe urzeczywistnienia. Powinniśmy zastanowić się, w jaki sposób możemy pomóc naszym uczniom odnaleźć się w coraz bardziej zautomatyzowanej rzeczywistości. Jedną z dróg jest nauka programowania, dzięki której lepiej zrozumiemy, jak działa cyfrowy świat, również ten, do którego wchodzi sztuczna inteligencja. ●

BIBLIOGRAFIA

1. Berghman L., Goossens D., Leus R. *Efficient solutions for Mastermind using genetic algorithms*, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.496.276&rep=rep1&type=pdf, dostęp 24.09.2019.
2. Borowiecka A., Olędzka K. *Wyszukiwanie, Mastermind i sztuczna inteligencja, Informatyka w Edukacji, Edukacja informatyczna a rozwój sztucznej inteligencji*, Wydawnictwo Naukowe UMK, Toruń 2019.
3. KhanAcademy, kurs Informatyka – Algorytmy, https://pl.khanacademy.org/computing/computer-science/algorithms, dostęp 24.09.2019.
4. Mastermind AI, https://github.com/ognjenvucko/mastermind-ai, dostęp 24.09.2019.
5. *Przewodnik po informatyce. Wersja dla nauczyciela*, https://bezkomputera.wmi.amu.edu.pl/ppi/teacher, dostęp 24.09.2019.