4rrsss

|  |  |  |  |
| --- | --- | --- | --- |
| agh_nzw_s_pl_1w_wbr_rgb_150ppi  **NAZWA WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**  Informatyka w sterowaniu i zarządzaniu  Głębokie uczenie i inteligencja obliczeniowa  Grupa 2 (śr 13:30 – 15:45)  *Układanie planu dla szkoły językowej* | | | |
| ***L.p.*** | **Członek** | **Numer albumu** | **Adres e-mail** |
| *1* | Patryk Chorąży | 402569 | pchorazy@student.agh.edu.pl |
| *2* | Rafał Kośla | 400332 | rkosla@student.agh.edu.pl |
| *3* | Artur Mzyk | 400658 | arturmzyk@student.agh.edu.pl |
| *4* | Joanna Nużka | 400561 | [joannanuzka@student.agh.edu.pl](mailto:joannanuzka@student.agh.edu.pl) |
| *5* | Adrian Poniatowski | 401346 | [adrianponiat@student.agh.edu.pl](mailto:adrianponiat@student.agh.edu.pl) |
| *6* | Wojciech Poniewierka | 402224 | wponiewierka@student.agh.edu.pl |

Spis treści

[1. Wstęp 3](#_Toc129799542)

[1.1. Cel projektu 3](#_Toc129799543)

[1.2. Założenia projektu 3](#_Toc129799544)

[2. Badany problem 3](#_Toc129799545)

[2.1. Opis problemu 3](#_Toc129799546)

[2.2. Przyjęte oznaczenia 4](#_Toc129799547)

[2.3. Postać rozwiązania 5](#_Toc129799548)

[2.4. Ograniczenia 5](#_Toc129799549)

[2.5. Funkcja celu 6](#_Toc129799550)

[2.6. Dane wejściowe 6](#_Toc129799551)

[2.7. Przegląd literatury 6](#_Toc129799552)

[3. Propozycja rozwiązania 6](#_Toc129799553)

[3.1. Algorytm 6](#_Toc129799554)

[3.1.1. Schemat podstawowego algorytmu 6](#_Toc129799555)

[3.1.2. Adaptacja algorytmu 6](#_Toc129799556)

[3.2. Pseudokod 6](#_Toc129799557)

[4. Aplikacja 6](#_Toc129799558)

[5. Eksperymenty 7](#_Toc129799559)

[6. Podsumowanie/wnioski 7](#_Toc129799560)

[7. Spis literatury 7](#_Toc129799561)

[8. Podział pracy 7](#_Toc129799562)

# Wstęp

## Cel projektu

Celem projektu jest zbudowanie aplikacji, która będzie za pomocą algorytmu genetycznego optymalizowała problem postawiony w tytule pracy, czyli „Układanie planu dla szkoły językowej”. Aplikacja zostanie rozbudowana o interfejs graficzny, który pozwoli na dynamiczną zmianę danych wejściowych oraz parametrów algorytmu. Dobór odpowiednich parametrów jest bardzo ważny z punktu widzenia optymalizacji, gdyż algorytm genetyczny daje jedynie rozwiązania bliskie optymalnemu. Odpowiednie dostrojenie – zazwyczaj  
w sposób empiryczny – może pozwolić na osiągnięcie znacznie lepszych rozwiązań niż  
w przypadku ustawienia losowej konfiguracji parametrów.

## Założenia projektu

Dane wejściowe wprowadzane są do aplikacji w postaci plików CSV.

# Badany problem

## Opis problemu

Rozważanym problemem jest ułożenie harmonogramu zajęć dla szkoły językowej. Jest to zagadnienie często podejmowane w innych pracach, zwłaszcza w kontekście uczelni czy szkół. Celem jest ustawienie lekcji w taki sposób, aby spełnione były wszystkie ograniczenia czasowe i te dotyczące zasobów zarówno ludzkich, jak i przestrzennych, oraz aby wartość przyjętej funkcji celu była jak najniższa. Ustalenie, że dana lekcja odbywa się w danym terminie, polega na powiązaniu ze sobą następujących danych:

* przedziałów czasowych formujących zajęcia,
* lektora,
* sali,
* uczniów o odpowiednim poziomie.

Szkoła językowa ma określoną liczbę uczniów, lektorów i sal. Są one niezależne od siebie. Liczba uczniów jest odczytywana z danych wejściowych – z uczniem powiązany jest jego poziom językowy. Liczba lektorów jest odczytywana z danych wejściowych – z lektorem są powiązane jego preferowane godziny pracy. Natomiast liczba sal jest pobierana z interfejsu graficznego.

Liczebność grup jest taka sama dla każdej z nich i pobierana jest z interfejsu graficznego. Liczba grup zależy od liczby przedziałów czasowych, w których jest otwarta szkoła, oraz od dostępności lektorów i sal.

Każdy uczeń i każda grupa mają określony poziom języka: 1, 2 lub 3. Do danej grupy mogą zostać zapisani uczniowie, których poziom różni się co najwyżej o 1 od poziomu tejże grupy. Za przypisanie ucznia do nieodpowiedniego poziomu jest przewidziana kara w funkcji celu. Kara w przypadku prawidłowego przypisania wynosi 0.

Dany uczeń może mieć tylko jedne zajęcia w ciągu tygodnia.

Dany lektor może uczyć dowolną grupę, gdyż jako wykształcony lingwistycznie człowiek potrafi doskonale język i poziom nie jest dla niego problemem.

Czas, w którym szkoła jest otwarta, jest podzielony na przedziały 15-minutowe. Pozwala to na elastyczne rozłożenie godzin zajęć i nie wymusza rozpoczynania ich o pełnej godzinie zegarowej. Przykładowo, jeśli zajęcia trwają godzinę, to zajmują one 4 takie przedziały. Długość – liczba przedziałów czasowych, które zajmują dane zajęcia – jest pobierana z interfejsu graficznego.

Dany lektor ma swoje preferowane godziny pracy. Funkcja celu uwzględnia karę za pracę lektora poza jego preferowanymi godzinami. Lektor nie potrzebuje przerwy pomiędzy zajęciami, gdyż czas funkcjonowania szkoły nie jest długi. Brak przerw nie powinien mocno wpływać na wydajność lektora.

Celem jest minimalizacja czasu spędzonego przez lektora w szkole językowej.

## Przyjęte oznaczenia

Zbudowano model problemu i zastosowano w nim następujące oznaczenia:

– rozwiązanie,

– liczba przedziałów czasowych (ang. *time slots*) na przestrzeni tygodnia, ,

– indeks przedziału czasowego, ,

– długość trwania zajęć (ang. *duration*), ,

– liczba lektorów (ang. *lectors*), ,

– liczba sal (ang. *rooms*), ,

– liczba zajęć (ang. *classes*) w przedziale czasowym , ,

– indeks zajęć w przedziale czasowym , ,

– indeks lektora powiązanego z przedziałem czasowym i zajęciami , ,

– indeks sali powiązanej z przedziałem czasowym i zajęciami , ,

– indeks lektora, ,

– zbiór bloków (ang. *blocks*), w których zajęcia prowadzi lektor ,

– zbiór tzw. okienek (ang. *windows*) lektora ,

– zbiór preferowanych (ang. *prefered*) przedziałów czasowych lektora ,

– zbiór bloków (ang. *blocks*), które są prowadzone w sali ,

– zbiór tzw. okienek (ang. *windows*) sali ,

– liczba grup (ang. *groups*), ,

– maksymalny rozmiar grupy (ang. *maximal group size*), ,

– liczba uczniów, ,

– liczba uczniów o znajomości (ang. *knowledge*) języka na poziomie 1, ,

– liczba uczniów o znajomości (ang. *knowledge*) języka na poziomie 2, ,

– liczba uczniów o znajomości (ang. *knowledge*) języka na poziomie 3, ,

– współczynnik kary za liczbę przedziałów czasowych przepracowanych przez lektora poza jego preferowanymi godzinami pracy, ,

– współczynnik kary za liczbę przedziałów czasowych będących tzw. okienkiem dla lektora, ,

– współczynnik kary za nieprawidłowe przypisanie ucznia do grupy językowej, .

## Postać rozwiązania

Z matematycznego punktu widzenia postać rozwiązania to wektor o stałej długości . Jego elementami są wektory o zmiennej długości . Z kolei ich elementami są piątki uporządkowane w postaci .

Z programistycznego punktu widzenia postać rozwiązania to -elementowa lista  
-elementowych krotek 5-elementowych.

## Ograniczenia

Liczba grup jest ograniczona przez liczbę uczniów i maksymalny rozmiar grupy :

.

Do danej grupy mogą zostać przypisani tylko uczniowie, których poziom różni się co najwyżej o 1 od poziomu tejże grupy:

Dana grupa musi mieć zajęcia w sąsiednich przedziałach czasowych, które formują razem blok zajęciowy, co jest intuicyjne:

gdzie

Dany uczeń może mieć zajęcia tylko raz w ciągu tygodnia:

Lektor może prowadzić tylko jedne zajęcia w tym samym czasie:

W jednej sali mogą się odbywać tylko jedne zajęcia w tym samym czasie:

## Funkcja celu

## Dane wejściowe

Dane wejściowe.

## Przegląd literatury

Przegląd literatury.

# Propozycja rozwiązania

Dokładny opis proponowanego rozwiązania (proponowanych rozwiązań).

## 3.1. Algorytm

### 3.1.1. Schemat podstawowego algorytmu

Wykorzystany w projekcie algorytm to algorytm genetyczny. Pozwala on na dążenie do lokalnego ekstremum, ale dalej możliwa jest większa eksploracja przestrzeni możliwych wyników w celu poszukiwania lepszych rozwiązań. Schemat jego działania można opisać w kilku krokach:

1. Utworzenie populacji początkowej – zbiór losowych rozwiązań problemu
2. Obliczenie dopasowania każdego osobnika w populacji, np. poprzez obliczenie wartości funkcji celu
3. Selekcja osobników do tworzenia kolejnej generacji – im wyższa wartość dopasowania, tym większa szansa na wybranie danego rozwiązania
4. Krzyżowanie – wybrane osobniki tworzą nowe poprzez wymianę części swoich genów
5. Mutacja – losowa zmiana niektórych genów, by dać szansę na rozwinięcie się nowych, korzystnych cech w populacji
6. Jeżeli nie osiągnięto zadowalających rezultatów i nie przekroczono maksymalnej liczby iteracji – powrót do kroku 2.

### 3.1.2. Adaptacja algorytmu

**Selekcje:**

Selekcje pozwalają na wybór odpowiednich osobników przy tworzeniu kolejnej populacji. Im lepsza wartość dopasowania danego rozwiązania, tym większą ma on szansę na wybranie. Zastosowano 3 podejścia przy problemie selekcji.

Selekcja best

W tym przypadku wybranymi do rozmnażania osobnikami są zawsze dwa najlepsze rozwiązania. Sposób ten jest eksperymentalny, ponieważ mocno ogranicza możliwość eksploracji przestrzeni rozwiązań.

Sposób generowania rozwiązań, sposób zakodowania rozwiązań (np. w algorytmie genetycznym sposób zrealizowania selekcji, krzyżowań i mutacji).

## 3.2. Pseudokod

Pseudokod.

# 4. Aplikacja

Sama aplikacja składa się z graficznego interfejsu użytkownika, dzięki któremu można wprowadzić dane startowe algorytmu oraz części odpowiedzialnej za sam algorytm. Do stworzenia GUI została użyta biblioteka PyQt5, natomiast część algorytmiczna została napisana w czystym Pythonie. Aby uruchomić aplikację wystarczy uruchomić skrypt pythonowy o nazwie main.py, następnie dobrać odpowiednie parametry i kliknąć przycisk Start.

# Eksperymenty

Badanie jakości algorytmu, różnych rozwiązań. Pokazać na przykładzie wyniki algorytmu.

# Podsumowanie/wnioski

Podsumowanie/wnioski.

# Spis literatury

# Podział pracy

Wkład każdego członka zespołu: etap realizacji i procentowy udział.