Projektowanie w optymalny sposób planu zajęć dla wydziału za pomocą algorytmu konstrukcyjnego.

Adrian Gacek, Ignacy Szkudelski

# Model zagadnienia – zamienić Sali na sali

Zagadnienie dotyczy problemu jak wygenerować optymalny plan zajęć dla wydziału.

Na wejściu algorytmu otrzymujemy:

* zestaw zajęć jakie muszą zostać przypisane do planu zajęć dla każdego kierunku
* zestaw sal dla każdych zajęć w jakich mogą się one odbywać
* przypisanego do każdych zajęć prowadzącego który będzie je prowadził
* listę parametrów za pomocą których możemy konfigurować przebieg algorytmu oraz funkcję celu

Algorytm konstrukcyjny próbuje w sposób inteligentny tak dobrać ułożenie zajęć aby zminimalizować funkcję celu przy jednoczesnym spełnieniu założeń:

* Żadne zajęcia nie mogą zachodzić na inne
* Pomiędzy każdymi zajęciami powinna być określona przerwa
* Nie powinny zostać żadne nieprzypisane zajęcia
* W jednej sali nie powinny odbywać się dwa zajęcia jednocześnie
* Zajęcia odbywają się w określonych ramach czasowych np. między 8 a 18.

Składowe funkcji celu:

* Minimalizacja czasu pomiędzy zajęciami
* Równomierność obciążenia na przestrzeni tygodnia
* Dni całkowicie wolne od zajęć
* Zajęcia mają zaczynać się jak naj wcześniej

Uproszczenia:

* Nie ma przedmiotów obieralnych
* Nie ma podziału na podgrupy
* Czasy przemieszczania się między salami są stałe dla wszystkich
* W ciągu tygodnia są dwa zajęcia z danego przedmiotu (ćwiczenia + wykład)
* Zajęcia mają określony czas trwania i nie są podzielne

# Wzory funkcji celu:

1. FO

Suma ze wszystkich dni sum czasów przerw między zajęciami pomniejszona o czas dojścia na dane zajęcia.

2. FD

Suma wag dni wolnych

3. FP

Dla każdego dnia dla każdych zajęć ich waga pory dnia

4. FR

Dla każdego dnia suma godzin zajęć które są ponad lub poniżej wartości IZS czyli średniej wartości w ciągu dnia z uwzględnieniem dnia wolnego. Jeżeli dzień jest wolny to jest to 0 dla danego dnia.

5. Funkcja celu

Współczynniki a,b,c,d to współczynniki wag.

Oznaczenia:

IZD - ilość zajęć w ciągu dnia

FO - funkcja zadowolenia z okienek

CP - jednostkowy czas przerwy

CD - czas dojścia

D - dzień tygodnia

Z - zajęcia

FD - funkcja zadowolenia z dni wolnych

wdw - waga wartości dnia wolnego

CZT - czas zajęć w ciągu tygodnia

wpd - waga zadowolenia odbywania zajęć o danej porze

FP - funkcja zadowolenia z pory zajęć

FR - funkcja zadowolenia z równomiernego obciążenia

CZD - czas zajęć w ciągu dnia

CDM - maksymalny czas trwania zajęć dla danej osoby w ciągu dnia

# Algorytm

Cały program składa się z trzech części (poza generowaniem danych):

1. Posortowanie listy zajęć
2. Preprocessing danych
3. Właściwy algorytm

### Sortowanie zajęć

Na tym etapie wprowadzona lista zajęć jest sortowana w taki sposób który zdaniem twórców naj bardziej zwiększy prawdopodobieństwo późniejszej optymalnej generacji planu.

W pierwszej kolejności przypisywane będą wykłady ponieważ przypisanie ich wymaga przypisania ich wielu grupom co mogło by być trudne lub niemożliwe gdyby miały one już wcześniej przypisane zajęcia.

Cała reszta zajęć dzielona jest na grupy do których dane zajęcia są przypisane – aby przydzielać zajęcia wszystkim grupom równomiernie.

Wewnątrz tych grup zajęcia sortowane są pod względem ilości dostępnych sal – łatwiej przypisać zajęcia które mogą się odbyć w większej ilości sal. Następnie taka lista dzielona jest na *n* sekcji i wewnątrz nich jest sortowana pod względem długości trwania – łatwiej przypisać krótsze zajęcia. Zmieniając ilość sekcji można wpływać na to czy większy udział będzie miała ilość dostępnych sal czy też czas trwania zajęć.

### Preprocessing danych

Ta część algorytmu dotyczy wyłącznie sal i ma na celu przygotowanie dodatkowej zmiennej „priorytetu danej sali”, który będzie decydował o kolejności wyboru sali podczas wybierania do której sali zostaną przypisane dane zajęcia. Jest on wyliczany na podstawie prawdopodobieństwa z jakim dana ilość zajęć odbędzie się w konkretnej Sali.

### Właściwy algorytm – pseudokod:

1. Do puki są jeszcze zajęcia do przypisania:

* Jeżeli nie -> zakończ algorytm
* Jeżeli tak -> weź kolejne z listy, kontynuuj

1. Sprawdź czy nie przekroczono maksymalnej liczby iteracji:

* Jeżeli tak -> zakończ algorytm
* Jeżeli nie -> zwiększ ilość iteracji o 1, kontynuuj

1. Dla danych zajęć
2. Wygeneruj wszystkie czasy w jakich zajęcia mogą się odbywać z uwzględnieniem jednostki czasu
3. Z tych czasów wybierz te w których jest dostępny prowadzący
4. Z pozostałych wybierz te które pasują wszystkim grupom (w przypadku wykładu)
5. Dla tych czasów sprawdź jaka będzie wartość funkcji celu grupy i prowadzącego jeżeli zajęcia zostały by do nich przypisane
6. Do puki został jeszcze jakiś możliwy czas:
7. Znajdź czas dla którego funkcja celu była najmniejsza
8. Sprawdź czy na liście dostępnych sal dla danych zajęć są takie którym pasuje wybrany czas:

* Jeżeli nie -> Kontynuuj pętlę z kroku 5
* Jeżeli tak -> Z tych sal wybierz tą która ma największy priorytet, przypisz zajęcia do planu danej Sali, prowadzącego i grup

1. Jeżeli nie udało się znaleźć żadnego terminu w którym można by przypisać dane zajęcia:

Zastosuj jedną z metod modyfikacji listy posortowanych zajęć

# Metody stosowane w razie niepowodzenia w przypisywaniu zajęć:

### Ignore

???

### Backtracking

???

### Reconstruction

???

### Replacing

???

# Parametry algorytmu:

1. Waga poszczególnych składowych funkcji celu – za pomocą tego parametru można regulować która ze składowych funkcji celu będzie bardziej brana pod uwagę. Parametry te skalują też wartości odpowiednich składowych. Jest to ważne z powodu trudności w oszacowaniu ich maksymalnych wartości.
2. Waga składowej całkowitej funkcji celu dla prowadzących – funkcja celu jest liczona osobno dla planu grupy, osobno dla planu prowadzącego. Wartość obliczona dla grupy mnożona jest razy ilość znajdujących się w niej studentów. Funkcja obliczona dla prowadzącego mnożona jest właśnie przez tą wagę.
3. Jednostka czasu jaka będzie używana w algorytmie – jednostka czasu określa jak długie będą przerwy pomiędzy zajęciami
4. Czas co jaki będą generowane dostępne dla danych zajęć czasy. Im mniejszy tym algorytm będzie działał dłużej.
5. Metoda cofania – zostały zaimplementowane 4 metody radzenia sobie w przypadku braku możliwości przypisania danych zajęć (opisane wyżej)
6. Ilość „przypisań” która zostaje cofnięta w razie nieudanego przypisania danych zajęć – parametr jednej z powyższych metod ignorowany w przypadku wybrania pozostałych
7. Maksymalna ilość iteracji – maksymalna ilość nieudanych „przypisań” po której algorytm kończy działanie
8. Ilość sekcji na ile zostaną podzielone zajęcia podczas wstępnego sortowania (według priorytetu przypisania) – im większa tym większe znaczenie ma ilość dostępnych sal dla danych zajęć, im mniejsza tym bardziej liczy się czas ich trwania. Opisane bardziej szczegółowo wyżej wyżej.

# Aplikacja

## Wymagania odnośnie uruchomienia

1. System Windows
2. Zainstalowany interpreter pythona w wersji 3.9
3. Zainstalowane moduły pythona: Importlib, Os, Pathlib, Typing, Abc, Copy, Math, Functools, csv, pdfscheduler, pandas

## Format danych wejściowych

Dane wejściowe są podawane w

1. Plikach .csv których kolumny dla poszczególnych części przedstawiają się następująco:

Grupy:

ID: int – niepowtarzalne

Group size: liczba studentów, int

Prowadzący:

ID: int – niepowtarzalne

Name: string

Sale:

ID: int – niepowtarzalne

Availability: int, ilość minut w jakich sala jest w sumie dostępna

Zajęcia:

ID: int – niepowtarzalne

Lecturer: id prowadzącego, int

Type: Lecture / Exercises, string

Duration: int, czas trwania zajęć

Rooms: lista id sal w jakich dane zajęcia mogą się odbywać w formie [int, int, …]

Groups: lista id grup w jakich dane zajęcia dotyczą

1. W pliku parameters.py podawane są w formie której szkic znajduje się w tym pliku parametry algorytmu

## Format danych wyjściowych

Wynikiem działania algorytmu są 3 foldery zawierające plany zajęć w plikach pdf osobno dla prowadzących, grup i sal.

Dodatkowo generowany jest raport z przebiegu algorytmu zawierający informacje:

* jakie dane wejściowe zostały wprowadzone
* Ile zajęć udało i nie udało się przypisać
* Końcowa wartość funkcji celu
* Wartości końcowe poszczególnych składowych wartości funkcji celu
* Czy powodem nieprzypisania zajęć był brak czasu w planie prowadzącego, grupy czy sal

## Funkcjonalności

* Zmienienie parametrów algorytmu
* Generacja danych / wprowadzenie własnych danych
* Parametryzacja generacji danych
* Generowanie raportu z przebiegu algorytmu
* Podgląd wygenerowanych planów

# Testy

# Podsumowanie

## Kierunki dalszego rozwoju

1. Rozwijając algorytm dalej można przede wszystkim postarać się zaimplementować obsługę wszystkich przypadków jakie występują w rzeczywistości. Na przykład: tygodnie parzyste i nieparzyste, zajęcia obieralne, czasy dojścia, niedostępność prowadzących
2. Kolejnym ważnym aspektem o jaki warto się zatroszczyć jest wymyślenie lepszych metod preprocessingu danych które pozwolą lepiej stworzyć plan
3. Można też stworzyć bardziej złożone metody radzenia sobie z brakiem możliwości przypisania danych zajęć
4. Można spróbować wymyślić metody które będą próbowały polepszyć rozwiązanie wygenerowane przez algorytm konstrukcyjny