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

Adrian Gacek,

Ignacy Szkudelski

# Model zagadnienia

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

Pomijanie zajęć, których nie da się przypisać.

### Backtracking

Cofanie sparametryzowanej ilości ostatnich wstawień, jako pierwsze zostaje następnie przypisanie poprzednio nieudanego.

### Reconstruction

Cofanie sparametryzowanej ilości ostatnich wstawień i losowa zmiana w kolejności przy powtórnym wstawianiu.

# 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

Zaimplementowany algorytm został przetestowany pod kątem wpływu zastosowanych parametrów oraz danych wejściowych na rozwiązanie. Testy zostały podzielone na trzy części:

* Test sprawności algorytmu
* Test mechanizmów poprawiających rozwiązania niedopuszczalne
* Test wpływu parametrów na jakość rozwiązania

Obserwacje i wnioski dotyczące testów będą opierać się na raportach generowanych przez każde wywołanie algorytmu.

## 1. Testy sprawności algorytmu

Testy mające na celu sprawdzenie rozwiązań dla problemów różnych rozmiarów i stopnia trudności. W przypadku problemów o mniejszej złożoności wyniki zostaną porównane z wynikami optymalnymi, gdy są one do wyznaczenia w prosty sposób.

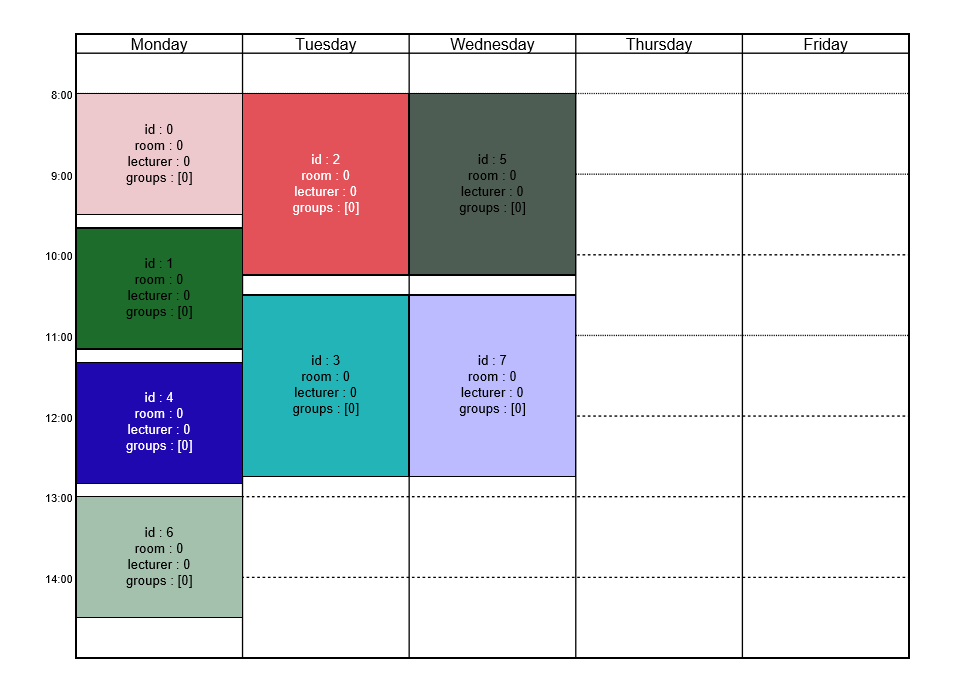
### 1.1 Problem prosty

Problem o rozwiązaniu łatwym do przewidzenia.

Własności danych wejściowych.

Liczba sal: 1  
Liczba prowadzących: 1  
Liczba grup: 1  
Liczba zajęć: 8

Rozwiązanie:

  
*Rys. 4.1 Rozwiązanie problemu prostego*

Właściwości rozwiązania:

Średni dzienny czas trwania zajęć: 5:00  
Średni czas rozpoczęcia zajęć: 8:00  
Średni czas zakończenia zajęć: 13:20  
Średnia liczba dni wolnych: 2  
Końcowa liczba przypisanych zajęć: 8  
Końcowa liczba zajęć bez przypisania: 0

Porównując otrzymane rozwiązanie z opisem z modelu zagadnienia można zauważyć, że jest ono bliskie optymalnemu – zajęcia są skupione w blokach, odbywają się w większości przed południem, jednocześnie nie są wstawiane pojedynczo, zachowując równomierność. Algorytm bardzo dobrze poradził sobie z rozwiązaniem prostego problemu.

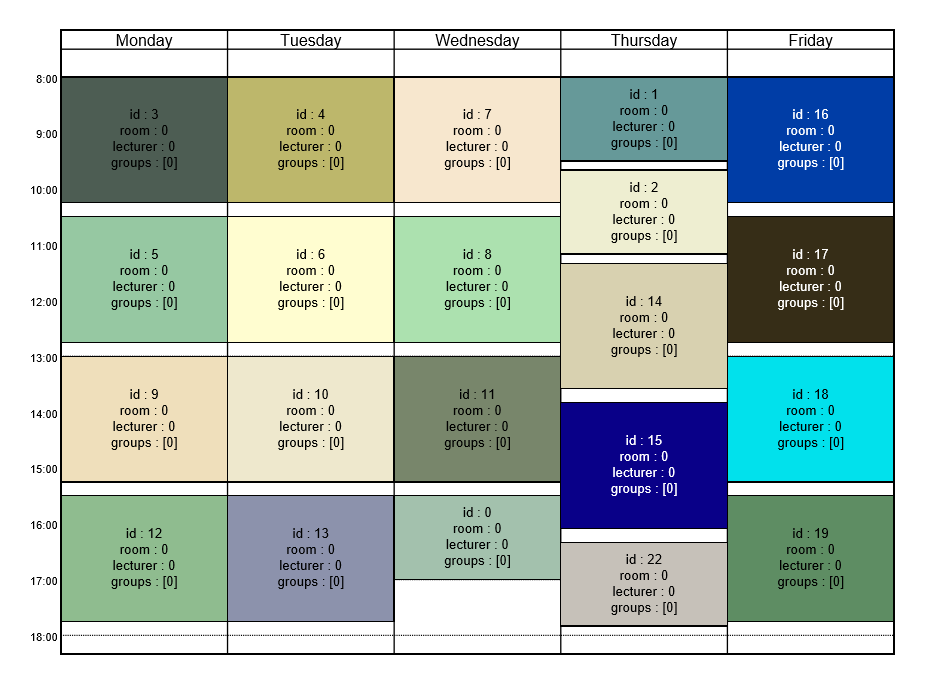
### 4.1.2 Problem prosty bez dopuszczalnego rozwiązania

Problem o rozwiązaniu łatwym do przewidzenia. Pozwala na sprawdzenie zachowania algorytmu dla problemów bez dopuszczalnego rozwiązania.

Własności danych wejściowych.

Liczba sal: 1  
Liczba prowadzących: 1  
Liczba grup: 1  
Liczba zajęć: 43

Rozwiązanie:

  
*Rys. 4.2 Rozwiązanie prostego problemu bez dopuszczalnego rozwiązania*

Właściwości rozwiązania:

Średni dzienny czas trwania zajęć: 8:51  
Średni czas rozpoczęcia zajęć: 8:00  
Średni czas zakończenia zajęć: 17:37  
Średnia liczba dni wolnych: 0  
Końcowa liczba przypisanych zajęć: 21  
Końcowa liczba zajęć bez przypisania: 22

Porównując otrzymane rozwiązanie z opisem z modelu zagadnienia można zauważyć, że zajęcia, które udało się przypisać są rozłożone w taki sposób, aby maksymalizować wykorzystanie dostępnych zasobów, minimalizując koszty ich rozszerzenia. Należy jednak zauważyć, że problem jest prosty, dla bardziej złożonych mogą pojawić się większe trudności, dlatego należałoby przeprowadzić więcej testów w tym kierunku.

### 4.1.3 Problem zbliżony do rzeczywistego

Problem oparty na wartościach dostępnych w Internecie[[1]](#footnote-1) oraz uzupełnione w sposób częściowo uporządkowany i częściowo losowy. Ma on za zadanie przetestować algorytm dla problemów zbliżonych do rzeczywistych oraz przygotować bazowe rozwiązanie dla kolejnych testów. Ponieważ rozwiązania kolejnych testów są zbyt duże, aby je analizować bezpośrednio, dalsze obserwacje będą opierać się na statystykach.

Własności danych wejściowych.

Liczba sal: 68  
Liczba prowadzących: 66  
Liczba grup: 92  
Liczba zajęć: 678

Właściwości rozwiązania:

Średni dzienny czas trwania zajęć: 4:21  
Średni czas rozpoczęcia zajęć: 8:46  
Średni czas zakończenia zajęć: 13:47  
Średnia liczba dni wolnych: 0  
Końcowa liczba przypisanych zajęć: 678  
Końcowa liczba zajęć bez przypisania: 0  
Wartość funkcji celu: 32.96  
Wartość składowych funkcji celu: FO: 4.96, FD 9.12, FP: 8.04, FR: 10.85

Algorytm podołał przypisaniu wszystkich zajęć. Można zauważyć, że czas trwania zajęć + średni czas zakończenia zajęć jest zbliżony do średniego czasu zakończenia, co oznacza, że zajęcia, były wstawiane równomiernie na przestrzeni tygodnia, bliżej godzin porannych, czego należałoby od niego oczekiwać.

### 4.1.4 Problem o znacznej złożoności

Problem oparty na danych z punktu 4.1.3, jednak zostały rozszerzone, aby przetestować działanie algorytmu dla złożonego problemu, który nie ma rozwiązania, lub jest ono bardzo trudne do osiągnięcia.

Własności danych wejściowych.

Liczba sal: 68  
Liczba prowadzących: 66  
Liczba grup: 92  
Liczba zajęć: 1147

Właściwości rozwiązania:

Średni dzienny czas trwania zajęć: 6:08  
Średni czas rozpoczęcia zajęć: 8:53  
Średni czas zakończenia zajęć: 16:19  
Średnia liczba dni wolnych: 0  
Końcowa liczba przypisanych zajęć: 1077  
Końcowa liczba zajęć bez przypisania: 70  
Wartość funkcji celu: 31.68  
Wartość składowych funkcji celu: FO: 10.37, FD 0.80, FP: 11.54, FR: 8.96

Porównując statystyki średnich czasów powyższego rozwiązania można zauważyć, że problem zawierający wiele zależności utrudnił znalezienie lepszego rozwiązania i zajęcia, jednak wartość funkcji celu ostatecznie jest niższa niż w poprzednim teście. Należy jednak zwrócić uwagę, że powyższe rozwiązanie jest niedopuszczalne: nie udało się przypisać wszystkich zajęć.

## 4.2. Testy poprawy rozwiązań niedopuszczalnych

Testy sprawdzające działanie mechanizmów zaimplementowanych w celu poprawienia rozwiązania niedopuszczalnego, tzn. takiego, w którym pojawią się zajęcia, których nie da się wstawić z powodu braku czasu uczestników. Dane wejściowe zostały przygotowane w taki sposób, aby niewielka (około 2%) liczba zajęć nie mogła być przypisana przy pierwszej próbie. Wyniki zostaną porównane z wynikiem ignorującym problem braku przypisania zajęć.

Własności danych wejściowych.

Liczba sal: 60  
Liczba prowadzących: 55  
Liczba grup: 92  
Liczba zajęć: 791

### 4.2.1 „Ignore”

Test będący bazą do badań metod poprawiających rozwiązania niedopuszczalne. Wszystkie zajęcia, których nie udało się przypisać są pomijane.

Parametry:

LECTURER\_WEIGHT 20  
FUN\_WEIGHTS (1, 1, 1, 1)   
SECTIONS\_AMOUNT: 3  
REASSIGN\_TYPE : 'ignore'   
UTIME: 10

Właściwości rozwiązania:

Liczba przypisań: 782  
Liczba nieudanych przypisań: 9  
Końcowa liczba przypisanych zajęć: 782  
Końcowa liczba zajęć bez przypisania: 9  
Wartość funkcji celu: 27.71

### 4.2.2 „Backtracking”

Testy metody cofającej sparametryzowaną ilość ostatnich wstawień i ustawiająca problematyczne zajęcia jako pierwsze w kolejce do wstawienia. Sprawdzony zostanie również wpływ parametrów STEP oraz MAX\_FAIL.

*Tabela 4.1 Wyniki dla metody „Backtracking” w zależności od parametrów STEP i MAX\_FAIL*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (STEP:MAX\_FAIL) | Liczba przypisań | Liczba nieudanych przypisań | Końcowa l. przypisanych zajęć | Końcowa l. nie przypisanych zajęć | Wartość funkcji celu |
| **(15:3)** | **1621** | **66** | **781** | **10** | **38.78** |
| (15:1) | 1621 | 66 | 781 | 10 | 38.78 |
| (15:5) | 1621 | 66 | 781 | 10 | 38.78 |
| (3, 3) | 946 | 73 | 775 | 16 | 37.79 |
| (30, 3) | 2372 | 62 | 782 | 9 | 39.09 |

Algorytm „Backtracking” w przypadku zastosowanych danych wejściowych okazał się bezużyteczny, może to wynikać z dwóch powodów: pierwszy - metoda zawiera pewien błąd w założeniach lub implementacji, drugi – algorytm wstawień jest na tyle dobry, że znajduje w badanych przypadkach najlepsze (w kontekście ilości przypisanych zajęć) rozwiązania jako pierwsze i próba poprawy kończy się niepowodzeniem.

### 4.2.3 „Reconstruction”

Testy metody cofającej sparametryzowaną ilość ostatnich wstawień i dokonująca losowej zmiany w kolejce, przy ponownym ustawianiu w kolejce. Sprawdzony zostanie również wpływ parametrów STEP oraz MAX\_FAIL.

*Tabela 4.2 Wyniki dla metody „Reconstruction” w zależności od parametrów STEP i MAX\_FAIL*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (STEP:MAX\_FAIL) | Liczba przypisań | Liczba nieudanych przypisań | Końcowa l. przypisanych zajęć | Końcowa l. nie przypisanych zajęć | Wartość funkcji celu |
| **(15:3)** | **1187** | **36** | **782** | **9** | **27.53** |
| (15:1) | 1099 | 28 | 784 | 7 | 29.21 |
| (15:5) | 1187 | 36 | 782 | 9 | 29.33 |
| (3, 3) | 863 | 36 | 782 | 9 | 27.71 |
| (30, 3) | 1503 | 32 | 783 | 8 | 28.46 |

Algorytm „Reconstruction” poprawia rozwiązanie w pewnych przypadkach, jednak trudno wywnioskować z powyższych wyników jakiekolwiek zależności, dlatego należałoby przeprowadzić więcej testów w tym kierunku.

## 4.3. Testy wpływu parametrów na jakość rozwiązania

Testy sprawdzające wpływ parametrów algorytmu na jakość rozwiązania opisanego poprzez funkcję celu. Wszystkie testy zostały przeprowadzone na tych samych danych wejściowych. Jako pierwszy został przeprowadzony test dla parametrów bazowych. Następnie w każdym teście zmieniano jeden z parametrów i obserwowano wynik.

Własności danych wejściowych.

Liczba sal: 68  
Liczba prowadzących: 66  
Liczba grup: 92  
Liczba zajęć: 678

### 4.3.1 Test dla parametrów bazowych

Problem z rozwiązaniem dopuszczalnym, do którego będą porównywane kolejne testy.

Parametry:

LECTURER\_WEIGHT 20  
FUN\_WEIGHTS (1, 1, 1, 1)   
SECTIONS\_AMOUNT: 3  
REASSIGN\_TYPE : 'ignore'   
UTIME: 10

*Tabela 4.3 Wynik bazowy do porównania*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | L. przypisanych | L. nie przypisanych | F. celu | FO | FD | FP | FR |
| **--------** | **678** | **0** | **32.96** | **4.96** | **9.12** | **8.03** | **10.85** |

Właściwości rozwiązania:

Średni dzienny czas trwania zajęć: 4:21  
Średni czas rozpoczęcia zajęć: 8:46  
Średni czas zakończenia zajęć: 13:47  
Średnia liczba dni wolnych: 0

### 4.3.2 Priorytet sortowania

Poniżej znajdują się wyniki testów dla różnych wartości parametru SECTIONS.

*Tabela 4.4 Wyniki względem zmiany parametru SECTIONS*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | L. przypisanych | L. nie przypisanych | F. celu | FO | FD | FP | FR |
| **3** | **678** | **0** | **32.96** | **4.96** | **9.12** | **8.03** | **10.85** |
| 1 | 678 | 0 | 29.73 | 3.81 | 7.84 | 8.08 | 10.00 |
| 200 | 678 | 0 | 31.64 | 4.36 | 8.76 | 8.04 | 10.48 |

Powyższa tabela pokazuje, że brak jest zależności między jakością rozwiązania, a faktem zwiększenia priorytetu sortowania względem długości zajęć lub dostępności sal. Parametr ten należy zapewne dobierać empirycznie w zależności od postawionego problemu.

### 4.3.3 Jednostka czasu

Testy dla różnych wartości parametru definiującego jednostkę czasu.

*Tabela 4.5 Wyniki względem zmiany parametru UTIME*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | L. przypisanych | L. nie przypisanych | F. celu | FO | FD | FP | FR |
| **10** | **678** | **0** | **32.96** | **4.96** | **9.12** | **8.03** | **10.85** |
| 5 | 678 | 0 | 59.40 | 7.80 | 22.44 | 10.54 | 18.62 |
| 20 | 672 | 6 | 46.60 | 11.30 | 11.76 | 10.59 | 12.95 |

Zmiana jednostki czasu niesie za sobą problem dopuszczalności rozwiązania – zwiększając ten parametr zmniejsza się liczba wolnych miejsc do wstawienia zajęć.

### 4.3.4 Wpływ składowych funkcji celu

Funkcja celu jest złożona z ważonych funkcji składowych. Test ma za zadanie określenie wpływu relacji między wagami.

*Tabela 4.6 Wyniki względem zmiany wag części składowych funkcji celu*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| (FO, FD, FP, FR) | Śr. Czas początku | Śr. Czas końca | Śr. Czas trwania | Śr. L. dni wolnych | F. celu |
| (1,1,1,1) | 8:46 | 13:47 | 4:21 | 0 | 32.96 |
| (10,1,1,1) | 9:03 | 15:35 | 5:26 | 1 | 126.87 |
| (1,10,1,1) | 9:03 | 15:35 | 5:26 | 1 | 311.83 |
| (1,1,10,1) | 9:03 | 15:35 | 5:26 | 1 | 168.56 |
| (1,1,1,10) | 9:03 | 15:35 | 5:26 | 1 | 259.50 |

Zmiana wag funkcji pory zmienia sposób wstawiania zajęć, jednak są one ze sobą ściśle powiązane, dlatego zmiana wagi jednej z nich wpływa podobnie jak zmiana każdej innej.

# 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.

1. Strona internetowa zawierająca aktualizowany plan zajęć wydziału EAIiIB AGH: https://planzajec.eaiib.agh.edu.pl/ [↑](#footnote-ref-1)