Przegląd wybranych metod optymalizacyjnych na przykładzie planu lekcji.

Sławomir Skrzypczak

25 czerwca 2013

Spis treści

Podziękowania

Serdeczne wyrazy podziękowania dla Pana prof. dr hab. Jerzego Szymańskiego za okazaną cierpliwość, wyrozumiałość i pomoc, dzięki której moja praca magisterska zyskała na wartości. Dziękuję za wszystkie cenne wskazówki które wykorzystałem w swojej pracy.

Wstęp

Celem pracy jest zaimplementowanie, na przykładzie planu lekcji dziennej szkoły średniej, wybranych algorytmów optymalizacyjnych. Optymalizacja planu lekcji jest problemem związanym z zarządzaniem zasobami. Istnieje wiele algorytmów optymalizujących gospodarkę zasobami, jednak praktycznie nie są one stosowane przy tworzeniu planu lekcji. Zazwyczaj wynika to z faktu, iż szkoła nie jest placówką nastawioną na generowanie zysków, co oznacza, że sam plan jest sprawdzany najczęściej tylko pod kątem poprawności, natomiast nie jest badany pod kątem generowanych kosztów.

Stworzenie planu lekcji, jest w wielu placówkach oświatowych zadaniem trudnym, czasochłonnym oraz zadaniem na wykonanie którego jest niewiele czasu. Dodatkowo w szkołach średnich plan lekcji może być zmieniany około trzy razy w roku: na początku I semestru, II semestru, po odejściu klas maturalnych lub też w wyniku jakiegoś zdarzenia losowego.

Na rynku dostępnych jest kilka programów generujących plan lekcji polskiej szkoły średniej. Bardzo często zdarza się jednak że nie są one w stanie wygenerować całego planu spełniającego wszystkie podane założenia, podając np 80% ułożonego planu. Resztę, ręcznie, musi ułożyć użytkownik programu, samemu decydując które założenia należy spełnić a które pominąć. Dotychczas nie spotkałem się z żadną aplikacją badającą koszty związane z wykonaniem planu lekcji. Resztę musi ułożyć ręcznie użytkownik programu samodzielnie decydując, które założenia należy spełnić a które pominąć. W związku z powyższym celem tej pracy jest optymalizacja tego zagadnienie.

Praca składa się z siedmiu rozdziałów. W pierwszym z nich zostanie wyprowadzonych kilka modeli matematycznych planu lekcji. Tematem rozdziału drugiego będzie kwestia poprawności planu. Następnie interesować nas będzie kiedy dany plan może istnieć. Idealną sytuacją byłoby wyprowadzenie warunku dostatecznego istnienia planu lekcji, co niestety nie zostało osiągnięte. W rozdziale trzecim wyprowadzone zostały jedynie warunki konieczne. Kwestia tworzenia planu lekcji została poruszona w rozdziale piątym. W rozdziale szóstym została omówiona propozycja funkcji celu. Rozdział siódmy został poświęcony algorytmom optymalizacyjnym. Ostatni rozdział podsumowuje uzyskane wyniki teoretyczne oraz praktyczne.

Rozdział 1

Wyprowadzenie modelu matematycznego planu lekcji

1.1 Jak powstaje plan lekcji?

Stworzenie planu lekcji wymaga poznania odpowiedzi na następujące pytania:

- Ile będzie klas w szkole?
- Jaki będą miały profil?
- Jak będzie wyglądał przydział dla nauczycieli?
- Kiedy nauczyciele są dyspozycyjni?
- Jak wyposażonych sal wymagają nauczyciele do prowadzenia danego przedmiotu?

Ilość klas w szkole średniej zależy przede wszystkim od naboru w danym roku. Dlatego liczba klas pierwszych jest często niewiadomą. Niektórych planowanych klas nie można otworzyć z powodu braku chętnych, niekiedy zaś liczba chętnych do nauki danego zawodu lub w danym profilu jest dużo wyższa niż liczba oferowanych miejsc. Co prowadzi do utworzenia, zamiast jednej klasy pierwszej w danym zawodzie, dwóch. W klasach wyższych, z różnych przyczyn, liczba uczniów się zmniejsza. Konieczne jest więc połączenie równoległych klas. Dodatkowo niektóre zajęcia nie mogą się odbywać w zbyt licznych grupach. Dla przykładu: obecnie nauka języka nie może się odbywać w grupie liczącej ponad 24 osoby, dlatego klasy liczące więcej niż 24 uczniów, są dzielone na grupy. Dodatkowym problemem są dość częste zmiany w przepisach regulujących pracę szkoły.

Ostateczną liczbę uczniów danej klasy można ustalić dopiero po zakończeniu naboru oraz egzaminów poprawkowych. Co ma miejsce w ostatnich dniach wakacji. Dlatego na stworzenie planu lekcji pozostaje więc tylko kilka dni.

Kolejnym etapem jest tworzenie przydziału dla nauczycieli, czyli ustalić który nauczyciel będzie uczył w danej klasie danego przedmiotu. Należy to zrobić oczywiście zgodnie z ich kwalifikacjami. A także zadbać aby liczba przepracowanych godzin była bliska liczbie godzin etatowych. Nauczyciel pracujący na pół etatu oraz nauczyciel z dużą liczbą nadgodzin, zazwyczaj są pracownikami mniej wydajnymi.

Problem też stanowi dyspozycyjność nauczyciel. Wielu z nich pracuje w dwóch lub trzech szkołach na część etatu. Osoby tworzące plan różnych szkół, muszą ustalić pomiędzy sobą kiedy konkretny nauczyciel może prowadzić przydzielone mu lekcje w danej placówce.

Trzeba się też zastanowić nad przydziałem sal lekcyjnych. Dla kilku przedmiotów jest niezbędne prowadzenie zajęć w konkretnej sali np. lekcja wychowania fizycznego musi odbywać się w sali gimnastycznej a lekcja informatyki w sali wyposażonej w komputery. Należy też uwzględnić preferencje nauczycieli. Niektórzy z nich nie wyobrażają sobie prowadzenia zajęć w sali niewyposażonej wprojektor multimedialny lub inne pomoce dydaktyczne. Czasami takie wymagania narzuca specyfika nauczanego przedmiotu. Z drugiej strony, są też nauczyciele którym taki sprzęt do niczego nie jest potrzebny.

Dopiero posiadając powyższe informacje można przystąpić do tworzenia planu lekcji.

Jeszcze przed kilku (kilkunastu) laty najpopularniejszą metodą układania planu lekcji była "metoda" kołków. Osoba układająca plan, czyli dyrektor lub wicedyrektor szkoły, układała plan ręcznie przy pomocy specjalnej tablicy kołkowej, kierując się swoimi spostrzeżeniami, doświadczeniem poprzedników oraz specyfiką pracy danej szkoły. Zazwyczaj nieświadomie, każda z tych osób, stosowała swój algorytm heurystyczny.

Obecnie często jest stosowane różnego rodzaju oprogramowanie. Jednak, jak już wspominałem we wstępie, często nie jest ono w stanie wygenerować całego planu lekcji. Jest to oprogramowanie nastawione wyłącznie na generowanie planu, a nie na rozwiązanie problemu problemu optymalizacji. Dodatkowo, oprogramowania o którym mowa, jest stosunkowo niewiele (Plan lekcji OPTIVUM, Plan lekcji - nPlan), a algorytmy wykorzystane, w tych rozwiązaniach, do generowania planu nie są jawnie udostępniane. Są one własnościami konkretnych firm.

1.2 Model matematyczny

Celem tego rozdziału jest znalezienie użytecznego matematycznego modelu planu lekcji. Obecnie nie zajmujemy się zagadnieniem poprawności planu lekcji. Będzie to temat kolejnych rozdziałów. Aby to osiągnąć, zaprezentowanych zostanie kilka modeli matematycznych planu lekcji. Dla zachowania przejrzystości, sam model oraz elementy z nim związane będą dodatkowo opatrywane przymiotnikami " typu I", "typu II" itd.

1.2.1 Model typu pierwszego

Aby wymodelować plan lekcji należy wymodelować pojedynczą lekcję. Potrzebna jest więc jakaś forma reprezentacji nauczycieli, przedmiotów, sal lekcyjnych oraz czasu rozpoczęcia lekcji.

W szkołach średnich większość zajęć trwa 45 minut. Istnieje kilka wyjątków od tej reguły np. zajęcia praktyczne przeprowadzane w kilkugodzinnych blokach. Jednak zazwyczaj są one "przycięte" w planie do kilku następujących po sobie 45-minutowych lekcji. Stąd, dla uproszczenia modelu, można założyć że wszystkie typy zajęć można reprezentować jako 45-minutowe lekcje. W związku z czym, w modelu nie będzie nam potrzebny czas zakończenia lekcji.

Niech K jest skończonym zbiorem klas uczonych w danej szkole np.

$$K = \{1a, 1b, 2a, 2b, 2c, \dots\}$$

Niech N jest skończonym zbiorem nauczycieli uczących w danej szkole np.

$$N = \{Adamczak, Bury, \dots\}$$

Niech P jest skończonym zbiorem nauczanych przedmiotów np.

$$P = \{\text{język polski, język angielski, matematyka, } \dots\}$$

Niech S jest skończonym zbiorem sal które posiada dana szkoła np.

$$S = \{01, 02, 03, 11, 12, 13, 21, 22, \dots\}$$

Niech C jest skończonym zbiorem czasów rozpoczęcia lekcji np.

$$C = \{\text{poniedziałek } 8^{00}, \text{ poniedziałek } 8^{55}, \text{ poniedziałek } 9^{50} \dots\}$$

Definicja 1.1. *Lekcją l_1 typu I nazywać będziemy piątkę $\{k, n, p, s, c\}$ gdzie $k \in K$, $n \in N$, $p \in P$, $s \in S$, $c \in C$.*

Definicja 1.2. *Planem lekcji typu I (oznaczanym przez PL_1) nazywać będziemy dowolny niepusty skończony zbiór lekcji l_1.*

Jak można zauważyć, taka definicja nie zapewnia poprawności planu pod żadnym względem. Pozwala za to na popełnienie wielu błędów takich jak na przykład:

- prowadzenie lekcji przez tego samego nauczyciela jednocześnie w kilku klasach, $\exists_{\{k_1,n_1,p_1,s_1,c_1\},\{k_1',n_1',p_1',s_1',c_1'\}} k_1 \neq k_1' n_1 = n_1' c_1 = c_1'$
- prowadzenie kilku różnych lekcji w tej samej klasie w tym samym czasie, $\exists_{\{k_1,n_1,p_1,s_1,c_1\},\{k_1',n_1',p_1',s_1',c_1'\}} k_1 = k_1' n_1 \neq n_1' c_1 = c_1'$ lub $\exists_{\{k_1,n_1,p_1,s_1,c_1\},\{k_1',n_1',p_1',s_1',c_1'\}} k_1 = k_1' p_1 \neq p_1' c_1 = c_1'$
- prowadzenie lekcji dla kilku klas w tej samej sali, $\exists_{\{k_1,n_1,p_1,s_1,c_1\},\{k_1',n_1',p_1',s_1',c_1'\}} k_1 \neq k_1' s_1 = s_1' c_1 = c_1'$

Dodatkowym problem jest reprezentacja planu. Po ponumerowaniu klas, nauczycieli, przedmiotów, sal i czasów lekcję można przedstawić jako pięciowymiarowy wektor. Natomiast sam plan jest skończonym niepustym zbiorem lekcji o nieustalonej liczebności. Jeżeli $\overline{\overline{PL_1}} = n$ to plan można przedstawić jako macierz o wymiarach 5 na n.

Jeżeli dokonamy numeracji ze względu na porządek leksykograficzny (pierwszemu elementowi w tym porządku przyporządkowujemy 1, drugiemu 2 itd.), to będziemy mogli przedstawiać plan w formie macierzy.

Przykład 1.1.

- *Dla przykładowego zbioru klas* $K = \{1a, 1b, 2a\}$ *odpowiadać będzie zbiór* $\{1, 2, 3\}$
- *Dla przykładowego zbioru nauczyciel* $N = \{$*Adamczak, Bury, Ciszewski*$\}$ *odpowiadać będzie zbiór* $\{1, 2, 3\}$,
- *Dla przykładowego zbioru przedmiotów* $P = \{$*fizyka, język angielski, język polski, matematyka,*$\}$ *odpowiadać będzie zbiór* $\{1, 2, 3, 4\}$,
- *Dla przykładowego zbioru sal* $S = \{01, 02, 11\}$ *odpowiadać będzie zbiór* $\{1, 2, 3\}$,
- *Dla przykładowego zbioru czasów rozpoczęcia lekcji* $C = \{$*poniedziałek* 8^{00}, *poniedziałek* $8^{55}\}$ *odpowiadać będzie zbiór* $\{1, 2\}$.

Wedy przykładowy plan typu I

$$
\begin{array}{llllllll}
PL_1 = \{ & (\textit{poniedziałek } 8^{00}, & \textit{1a,} & \textit{Adamczak,} & \textit{język angielski,} & \textit{01),} & \\
& (\textit{poniedziałek } 8^{00}, & \textit{1b,} & \textit{Bury,} & \textit{język polski,} & \textit{02),} & \\
& (\textit{poniedziałek } 8^{55}, & \textit{1a,} & \textit{Bury,} & \textit{język polski,} & \textit{02),} & \\
& (\textit{poniedziałek } 8^{55}, & \textit{1b,} & \textit{Adamczak,} & \textit{język angielski,} & \textit{01),} & \\
& (\textit{poniedziałek } 8^{55}, & \textit{2a,} & \textit{Ciszewski,} & \textit{matematyka,} & \textit{11)} & \}
\end{array}
$$

Można przedstawić w formie tabeli:

Czas	*poniedziałek* 8^{00}	*poniedziałek* 8^{00}	*poniedziałek* 8^{55}
Klasa	*1a*	*1b*	*1a*
Nauczyciel	*Adamczak*	*Bury*	*Bury*
Przedmiot	*język angielski*	*język polski*	*język polski*
Sala	*01*	*02*	*02*

Czas	*poniedziałek* 8^{55}	*poniedziałek* 8^{55}	
Klasa	*1b*	*2a*	
Nauczyciel	*Adamczak*	*Ciszewski*	
Przedmiot	*język angielski*	*matematyka*	
Sala	*01*	*11*	

Tabela ta jest równoważna macierzy:

$$
\begin{array}{l|ccccc|}
\textbf{Czas} & 1 & 1 & 2 & 2 & 2 \\
\textbf{Klasa} & 1 & 2 & 1 & 2 & 3 \\
\textbf{Nauczyciel} & 1 & 2 & 2 & 1 & 3 \\
\textbf{Przedmiot} & 1 & 2 & 2 & 1 & 3 \\
\textbf{Sala} & 1 & 2 & 2 & 1 & 3
\end{array}
$$

1.2.2 Model typu drugiego

Wadą modelu typu I jest duża oraz nieustalona dokładnie liczba zmiennych potrzebna do reprezentacji planu.

Spróbujmy zapisać plan w innej, oszczędniejszej reprezentacji. Pomoże nam w tym potraktowanie planu jako ciągu a nie jako zbioru lekcji. Poprzez miejsce lekcji w ciągu wyrażona będzie klasa oraz czas odbywania lekcji. Chcemy też aby liczba lekcji w planie była stała. Jeżeli w szkole jest k klas i lekcje mogą się odbywać w c czasach to maksymalnie może się odbyć kc lekcji. Np. niech pierwsze k lekcji to lekcje rozpoczynające się o 8^{00}, następne k to lekcję rozpoczynające się o 8^{55} itd. Dalej każdą grupę k lekcji uszeregujemy według klas, czyli 1 lekcja z danej grupy niech odbywa

się w klasie 1a, druga w 1b trzecia w 2a itd. Aby o każdej porze, klasie można przyporządkować jakąś lekcję, wprowadzimy lekcję bez przydzielonej sali, nauczyciela i przedmiotu. Wprowadzimy również pustą lekcję. W szczególności, gdy klasy nie będzie w szkole, będzie reprezentowana przez tą lekcję. Dzięki temu zabiegowi, plan dla szkoły o k klasach oraz c czasach rozpoczęcia lekcji, zawsze będzie można zapisać przy pomocy kc lekcji. W tym celu należy rozszerzyć każdy ze zbiorów N, P, S o specjalny pusty element. Element ten dobierzemy tak aby w porządku leksykograficznym używanym do przenumerowania elementów zbioru był on na początku.
I tak niech $N' = N \cup \{\epsilon\}$ (pusty napis), $P' = P \cup \{\epsilon\}$, $S' = S \cup \{0\}$. Tak więc:

Definicja 1.3. *Lekcją typu II (oznaczana przez l_2) nazywać będziemy trójkę $\{n, p, s\}$ gdzie $n \in N'$, $p \in P'$, $s \in S'$.*

Definicja 1.4. *Niech $\overline{\overline{K}} = k$ oraz $\overline{\overline{C}} = c$. Planem lekcji typu II (oznaczanym przez $PL_2(K, C)$ nazywać będziemy ciąg lekcji l_2 mający kc-elementów.*

Przykład 1.2. *Spróbujemy zapisać plan lekcji typu PL_2 w sposób analogiczny planowi z przykładu 1.1.*

- *dla zbioru klas $K = \{1a, 1b, 2a\}$ jest równoważny zbiorowi $\{1, 2, 3\}$ (pierwszemu elementowi w porządku leksykograficznym, przyporządkowujemy 1, drugiemu 2 itd.),*
- *dla zbioru nauczyciel $N' = \{\epsilon$,Adamczak, Bury, Ciszewski $\}$ jest równoważny zbiorowi $\{0, 1, 2, 3\}$ (pierwszemu elementowi w porządku leksykograficznym, przyporządkowujemy 0, drugiemu 1 itd.),*
- *dla zbioru przedmiotów $P' = \{\epsilon$,fizyka, język angielski, język polski, matematyka,$\}$ jest równoważny zbiorowi $\{0, 1, 2, 3, 4\}$ (pierwszemu elementowi w porządku leksykograficznym, przyporządkowujemy 0, drugiemu 1 itd.),*
- *dla zbioru sal $S' = \{0, 01, 02, 11\}$ jest równoważny zbiorowi $\{0, 1, 2, 3\}$ (pierwszemu elementowi w porządku leksykograficznym, przyporządkowujemy 0, drugiemu 1 itd.),*
- *dla zbioru czasów rozpoczęcia lekcji $C = \{$poniedziałek 8^{00},poniedziałek $8^{55}\}$ jest równoważny zbiorowi $\{1, 2\}$ (pierwszemu elementowi w porządku leksykograficznym, przyporządkowujemy 1, drugiemu 2 itd.),*

Wtedy plan:

$$PL'_2 = \Big(\begin{array}{lll} (\textit{Adamczak}, & \textit{język angielski}, & 01), \\ (\textit{Bury}, & \textit{język polski}, & 02), \\ (\epsilon, & \epsilon, & 0), \\ (\textit{Bury}, & \textit{język polski}, & 02), \\ (\textit{Adamczak}, & \textit{język angielski}, & 01), \\ (\textit{Ciszewski}, & \textit{matematyka}, & 11) \end{array} \Big)$$

jest odmienny od planu

$$P''_{l2} = \Big(\begin{array}{lll} (\textit{Adamczak}, & \textit{język angielski}, & 01), \\ (\epsilon, & \epsilon, & 0), \\ (\textit{Bury}, & \textit{język polski}, & 02), \\ (\textit{Bury}, & \textit{język polski}, & 02), \\ (\textit{Adamczak}, & \textit{język angielski}, & 01), \\ (\textit{Ciszewski}, & \textit{matematyka}, & 11) \end{array} \Big)$$

ponieważ różnią się one, pomiędzy sobą, na drugim i trzecim miejscu. Plan PL'_2 *można przedstawić w formie tabeli:*

Czas	*poniedziałek* 8^{00}	*poniedziałek* 8^{00}	*poniedziałek* 8^{00}
Klasa	*1a*	*1b*	*2a*
Nauczyciel	*Adamczak*	*Bury*	ϵ
Przedmiot	*język angielski*	*język polski*	ϵ
Sala	*01*	*02*	*0*
Czas	*poniedziałek* 8^{55}	*poniedziałek* 8^{55}	*poniedziałek* 8^{55}
Klasa	*1a*	*1b*	*2a*
Nauczyciel	*Bury*	*Adamczak*	*Ciszewski*
Przedmiot	*język polski*	*język angielski*	*matematyka*
Sala	*02*	*01*	*11*

Tabela ta jest równoważna macierzy:

$$\begin{array}{l|cccccc|} \textbf{Czas} & 1 & 1 & 1 & 2 & 2 & 2 \\ \textbf{Klasa} & 1 & 2 & 3 & 1 & 2 & 3 \\ \textbf{Nauczyciel} & 1 & 2 & 0 & 2 & 1 & 3 \\ \textbf{Przedmiot} & 1 & 2 & 0 & 2 & 1 & 3 \\ \textbf{Sala} & 1 & 2 & 0 & 2 & 1 & 3 \end{array}$$

W wierszu czas widać że mamy trzy (k=3) jedynki następnie trzy dwójki. Natomiast w wierszu odpowiedzialnym za klasy mamy dwa (c=2) razy powtórzony ciąg 1,2,3.

Dlatego do zapisu planu mającego trzy klasy oraz dwie godziny rozpoczęcia zajęć wystarczy macierz o wymiarach 3 *na* $3 \cdot 2$*.*

Nauczyciel	*1*	*2*	*0*	*2*	*1*	*3*
Przedmiot	*1*	*2*	*0*	*2*	*1*	*3*
Sala	*1*	*2*	*0*	*2*	*1*	*3*

1.2.3 Model typu trzeciego

W planie lekcji typu II, dla szkoły o k klasach i c czasach rozpoczęcia lekcji istnieje kc lekcji. Samą lekcję można interpretować jako trójwymiarowy wektor, więc do zapisu planu potrzebujemy $3kc$ liczb. Mała szkoła średnia ma około 10 klas oraz $40 = 5 * 8$ czasów rozpoczęcia lekcji. Do zapisu jej planu potrzeba więc $3 * 10 * 40 = 1200$ liczb. Taki plan można rozpatrywać jako 1200-wymiarowy wektor, lub macierz o wymiarach 400 na 3. Tak duża ilość wymiarów, bardzo utrudnia implementacje algorytmów. Postaram się ją zmniejszyć, stosując funkcje pozycyjne.

Definicja 1.5. *Niech S będzie zbiorem skończonym i $\overline{\overline{S}} = n$ gdzie $n \in \mathbb{N}$, oraz niech $\prec$ będzie relacją porządkującą określoną na elementach zbioru S. Funkcją pozycyjną nazywać będziemy bijekcję $rank : S \to \{0, 1, \ldots, n-1\}$ określającą całkowity porządek na zbiorze S. Funkcję pozycyjną działającą na $s \in S$ oznaczać będziemy $rank(S)$.*

$$\forall_{s,t \in S} s \prec t \Leftrightarrow rank(s) < rank(t)$$

W planie typu II pierwszy wiersz macierzy odpowiada za przyporządkowanie nauczycieli, drugi za przyporządkowanie przedmiotów a trzeci za przyporządkowanie sal. Każdy z tych trzech wierszy można potraktować jako k-elementową wariację z powtórzeniami pewnego zbioru n-elementowego. W przypadku pierwszego wiersza n będzie równe liczbie nauczycieli uczących w szkole powiększonej o 1, w przypadku wiersza drugiego n będzie równe liczbie nauczanych przedmiotów powiększonej o 1 a w przypadku trzecim n będzie równe liczbę sal powiększonej o 1. Po zastosowaniu funkcji $rank$ do każdego wiersza z osobna, plan będziemy mogli przedstawić jako trójwymiarowy wektor.

Definicja 1.6. *Jeżeli istnieje funkcja pozycyjna zadana na zbiorze S to istnieje dokładnie jedna funkcja antypozycyjna (oznaczamy $anrank(i, k, n)$ gdzie $i, k, n \in N$) będąca funkcją odwrotną do $rank$.*

$$\forall_{s, \in S} \forall_{i \in \{0,1,2,\ldots,n-1\}} anrank(i) = s \Leftrightarrow rank(s) = i$$

Interesować nas obecnie będzie postać funkcji *rank* i *anrank* dla k-elementowych wariacji zbioru n-elementowego. Dlatego funkcje *rank* i *anrank* zostaną opatrzone dodatkowymi argumentami, mianowicie k oraz n. Wariacje, o których mowa, uporządkowane są w sposób leksykograficzny. Niech $a_0, a_1, \ldots, a_{k-1} \in \{0, \ldots, n-1\}$. Wariację k-elementową zbioru n-elemntowego, oznaczaną przez s będziemy zapisywać w postaci

$$s = (a_{k-1}, a_{k-2}, \ldots, a_0)$$

Wtedy:

$$rank(s, k, n) = \sum_{i=0}^{i=k-1} a_i n^i$$

Natomiast

$$anrank(w, k, n) =$$

$$= \Big(partanrank(k-1, w, k, n), partanrank(k-2, w, k, n), \ldots, partanrank(0, w, k, n)\Big)$$

gdzie funkcja $partanrank(i, w, k, n)$ zwraca element a_i wariacji k-elementowej zbioru n-elementowego z wariancji o $rank(s, k, n) = w$.

$$partanrank(i, w, k, n) = \begin{cases} \frac{w-(w \mod n^i)}{n^i} & dla \quad i = k-1 \\ \frac{(w \mod n^{i+1})-(w \mod n^i)}{n^i} & dla \quad 0 < i < k-1 \\ w \mod n & dla \quad i = 0 \end{cases}$$

Przykład 1.3. *Niech dany będzie plan szkoły która posiada 3 klasy, 3 czasy rozpoczęcia lekcji, 4 nauczycieli, 5 przedmiotów oraz 3 sale. Wtedy* $\overline{\overline{N'}} = 5$, $\overline{\overline{P'}} = 6$ *oraz* $\overline{\overline{S'}} = 4$.

Nauczyciel	*1*	*2*	*3*	*3*	*2*	*1*	*3*	*0*	*0*
Przedmiot	*1*	*2*	*3*	*3*	*2*	*4*	*3*	*0*	*0*
Sala	*2*	*1*	*3*	*3*	*1*	*2*	*3*	*0*	*0*

- *Obliczam rank z pierwszego wiersza oznaczającego przydział nauczycieli:*

$$rank((1, 2, 3, 3, 2, 1, 3, 0, 0), 9, 5) =$$

$$= 1 \cdot 5^8 + 2 \cdot 5^7 + 3 \cdot 5^6 + 3 \cdot 5^5 + 2 \cdot 5^4 + 1 \cdot 5^3 + 3 \cdot 5^2 + 0 \cdot 5^1 + 0 \cdot 5^0 =$$

$$= 390625 + 156260 + 46875 + 9375 + 1250 + 125 + 75 = 604575$$

- *Obliczam rank z drugiego wiersza oznaczającego przydział przedmiotów:*

$$rank((1, 2, 3, 3, 2, 4, 3, 0, 0)9, 6) =$$

$$= 1 \cdot 6^8 + 2 \cdot 6^7 + 3 \cdot 6^6 + 3 \cdot 6^5 + 2 \cdot 6^4 + 4 \cdot 6^3 + 3 \cdot 6^2 + 0 \cdot 6^1 + 0 \cdot 6^0 =$$

$$= 2406348$$

- *Obliczam rank z trzeciego wiersza oznaczającego przydział sal:*

$$rank((2,1,3,3,1,2,3,0,0)9,4) =$$

$$= 2 \cdot 4^8 + 2 \cdot 4^7 + 3 \cdot 4^6 + 3 \cdot 4^5 + 1 \cdot 4^4 + 2 \cdot 4^3 + 3 \cdot 4^2 + 0 \cdot 4^1 + 0 \cdot 4^0 =$$

$$= 163248$$

Tak więc dany plan można zapisać w postaci wektora $(604575, 2406348, 163248)$.

Przykład 1.4. *Obliczmy* $anrank(604575, 9, 5)$. *Czyli odtworzymy pierwszy wiersz naszego planu. Jest to wiersz odpowiadający za przyporządkowanie nauczycieli. W tym celu należy obliczyć:*

$$partanrank(0, 604575, 9, 5) = 604575 \mod 5 = 0$$

$$partanrank(1, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 25) - (604575 \mod 5)}{5} = \frac{0-0}{5} = 0$$

$$partanrank(2, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 125) - (604575 \mod 25)}{25} = \frac{75-0}{25} = 3$$

$$partanrank(3, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 625) - (604575 \mod 125)}{125} = \frac{200-75}{125} = 1$$

$$partanrank(4, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 5^5) - (604575 \mod 5^4)}{5^4} = \frac{1450-200}{5^4} = \frac{1250}{625} = 2$$

$$partanrank(5, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 5^6) - (604575 \mod 5^5)}{5^5} = \frac{10825-1450}{5^5} = 3$$

$$partanrank(6, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 5^7) - (604575 \mod 5^6)}{5^6} = \frac{57700-1450}{5^6} = 3$$

$$partanrank(7, 604575, 9, 5) =$$
$$= \frac{(604575 \mod 5^8) - (604575 \mod 5^7)}{5^7} = \frac{213950-57700}{5^7} = 2$$

$$partanrank(8, 604575, 9, 5) =$$
$$= \frac{604575 - (604575 \mod 5^8)}{5^4} = \frac{604575-213950}{5^8} = \frac{390625}{5^8} = 1$$

Tak więc $anrank(604575, 9, 5) = (1, 2, 3, 3, 2, 1, 3, 0, 0)$.

Analogicznie można obliczyć $anrank(2406348, 9, 6)$. *Otrzymamy wiersz drugi opisujący przedmioty, a* $anrank(163248, 9, 4)$ *da w wyniku wiersz trzeci opisujący sale.*

1.2.4 Model typu czwartego

Tworząc obecnie opisywany model wykorzystany zostanie fakt posiadania przydziału pracy nauczyciela. Wiemy ile każdy nauczyciel musi mieć lekcji ustalonego przedmiotu w danej klasie. Wiemy że pan Kowalski w klasie 1a ma do przeprowadzenie trzy lekcje języka polskiego i jedną lekcję wiedzy o kulturze. Natomiast w 2a, trzy lekcje języka polskiego itd. Uszeregujemy lekcje typu II $\{n, p, s\}$ gdzie $n \in N'$, $p \in P'$, $s \in S'$ w porządku leksykograficznym według wartości n, p, oraz s. Następnie każdej lekcji przyporządkujemy czas i salę. W modelu tym nie jest potrzebna reprezentacja pustej lekcji ("okienka").

Definicja 1.7.
Niech K jest skończonym zbiorem klas uczonych w danej szkole.
Niech N jest skończonym zbiorem nauczycieli uczących w danej szkole.
Niech P jest skończonym zbiorem nauczanych przedmiotów.
Niech $n \in N$, $p \in P$, $k \in K$.
Wtedy przydziałem Π nazywamy ciąg trójek postaci (n, p, k).

Definicja 1.8.
Niech K jest skończonym zbiorem klas uczonych w danej szkole.
Niech N jest skończonym zbiorem nauczycieli uczących w danej szkole.
Niech P jest skończonym zbiorem nauczanych przedmiotów.
Niech $n \in N$, $p \in P$, $k \in K$.
Wtedy przydziałem uporządkowanym Π' nazywamy przydział powstały poprzez uporządkowanie przydziału Π w sposób leksykograficzny, kolejno względem wartości n, p oraz k.

Przykład 1.5. *Niech dany będzie przydział uporządkowany*

$$\Pi = \Big(\begin{array}{lll} (\textit{Adamczak}, & \textit{język angielski}, & \textit{1a}), \\ (\textit{Adamczak}, & \textit{język angielski}, & \textit{1b}), \\ (\textit{Bury}, & \textit{język polski}, & \textit{1a}), \\ (\textit{Bury}, & \textit{język polski}, & \textit{1b}), \\ (\textit{Ciszewski}, & \textit{matematyka}, & \textit{2a}) \end{array} \Big)$$

mający trzech nauczycieli, trzy klasy oraz trzy przedmioty. Na jego podstawie może powstać przykładowy plan

$$P_l = \Big(\begin{array}{lllll} \textit{(Adamczak,} & \textit{język angielski,} & \textit{1a,} & \textit{01,} & \textit{poniedziałek } 8^{00}\textit{),} \\ \textit{(Adamczak,} & \textit{język angielski,} & \textit{1b,} & \textit{01,} & \textit{poniedziałek } 8^{55}\textit{),} \\ \textit{(Bury,} & \textit{język polski,} & \textit{1a,} & \textit{02,} & \textit{poniedziałek } 8^{55}\textit{),} \\ \textit{(Bury,} & \textit{język polski,} & \textit{1b,} & \textit{02,} & \textit{poniedziałek } 8^{00}\textit{),} \\ \textit{(Ciszewski,} & \textit{matematyka,} & \textit{2a,} & \textit{11,} & \textit{poniedziałek } 8^{55}\textit{)} \end{array} \Big)$$

mający dwa czasy rozpoczęcia lekcji oraz trzy sale. Dany plan można przedstawić w formie tabeli:

Nauczyciel	*Adamczak*	*Adamczak*	*Bury*	*Bury*	*Ciszewski*	
Przedmiot	*język angielski*	*język angielski*	*język polski*	*język polski*	*matematyka*	
Klasa	*1a*	*1b*	*1a*	*1b*	*2a*	
Czas	*poniedziałek* 8^{00}	*poniedziałek* 8^{55}	*poniedziałek* 8^{55}	*poniedziałek* 8^{00}	*poniedziałek* 8^{55}	
Sala	*01*	*01*	*02*	*02*	*11*	

Tabela ta jest równoważna macierzy:

$$\begin{array}{l|ccccc|} \textbf{\textit{Nauczyciel}} & 1 & 1 & 2 & 2 & 3 \\ \textbf{\textit{Przedmiot}} & 1 & 1 & 2 & 2 & 3 \\ \textbf{\textit{Klasa}} & 1 & 2 & 1 & 2 & 3 \\ \textbf{\textit{Czas}} & 1 & 2 & 2 & 1 & 2 \\ \textbf{\textit{Sala}} & 1 & 1 & 2 & 2 & 3 \end{array}$$

Pierwsze trzy wiersze można będzie odtworzyć na podstawie przydziału nauczycieli. Dlatego do zapisu planu konieczne są tylko dwa ostatnie wiersze. Plan lekcji można zapisać w następującej postaci:

$$\begin{array}{l|ccccc|} \textbf{\textit{Czas}} & 1 & 2 & 2 & 1 & 2 \\ \textbf{\textit{Sala}} & 1 & 1 & 2 & 2 & 3 \end{array}$$

Niech S jest skończonym zbiorem sal które posiada dana szkoła. Niech C jest skończonym zbiorem czasów rozpoczęcia lekcji.

Definicja 1.9. *Niech dany jest uporządkowany przydział* Π' *mający* π *elementów, niech* $c \in C$ *oraz* $s \in S$*. Planem lekcji typu IV nazywamy* π*-elementowy ciąg par postaci* (c, s)*.*

Widać że plan z powyższego przykładu spełnia definicję planu typu IV. Dodatkowo plan ten można przedstawić jako dwuwierszową macierz. Wiersze te można, analogicznie do modelu III, potraktować jako wariację π-elementową odpowiednio $\overline{\overline{S}}$ i $\overline{\overline{C}}$ elementów. Co umożliwia przedstawienie planu typu IV w formie dwuwierszowego wektora.

Rozdział 2

Poprawność planu lekcji

W tym rozdziale postaramy się zdefiniować pojęcie poprawności plan lekcji. Jak należy rozumieć poprawność planu? Chcemy aby plan:

1. Spełniał cały dany przydział.

2. Nie miał "kolizji", czyli nie dochodziło do sytuacji w której:

   (a) nauczyciel ma do przeprowadzenia dwie różne lekcje w tym samym czasie,

   (b) nauczyciel pracuje w czasie w którym jest dyspozycyjny,

   (c) klasa ma, w tym samym czasie, lekcje prowadzone przez dwóch różnych nauczycieli,

   (d) w danej sali, w tym samym czasie, są zaplanowane dwie różne lekcje.

3. Klasy miały lekcje przeprowadzane jedna po drugiej, czyli patrząc z perspektywy klasy, bez "okienek".

Można stwierdzić że plan p typu IV spełnia dany π-elementowy przydział Π. Wystarczy że plan p jest ciągiem π-elementowym.

Konflikt przedstawione w podpunkcie $2a$ można opisać matematycznie w następujący sposób:

Definicja 2.1.
Niech p jest planem typu IV realizującym przydział Π. Mówimy że nauczyciel ma konflikt na miejscu i-tym i j-tym gdy $i, j \leqslant \pi, i \neq j$, $\Pi_i = (t_i, d_i, q_i)$, $\Pi_j = (t_j, d_j, q_j)$, $p_i = (c_i, s_i)$, $p_j = (c_j, s_j)$ oraz

$$t_i = t_j \wedge c_i = c_j$$

W spełnieniu warunku $2b$ niezbędna będzie macierz obrazująca dyspozycyjność nauczyciela, mająca tyle kolumn ile jest czasów rozpoczęcia lekcji oraz tyle wierszy ilu jest nauczycieli. Jej elementami będzie 0 gdy nauczyciel nie może pracować w danym czasie oraz 1 gdy może prowadzić lekcje. Macierz tą będziemy oznaczać przez $DN_{n\ c}$. Jest również ona pomocna do sprawdzenia warunku $2a$. Wystarczy w tym celu, dla każdej lekcji, sprawdzić czy w wierszu o numerze odpowiadającym nauczycielowi i kolumnie odpowiadającej czasowi prowadzenia lekcji jest wartość 1. Jeżeli tak jest, to należy ją zmienić na 0, jeżeli natomiast trafimy na 0 oznacza to znalezienie konfliktu.

Podobnie należy potraktować problem dostępności sal lekcyjnych.

Definicja 2.2.
Niech p jest planem typu IV realizującym przydział Π. Mówimy, że sala ma konflikt na miejscu i-tym i j-tym gdy $i, j \leqslant \pi, i \neq j$, $p_i = (c_i, s_i)$, $p_j = (c_j, s_j)$ oraz

$$s_i = s_j \wedge c_i = c_j$$

Utworzymy też macierz analogiczną do $DN_{n\ c}$. Będzie ona miała tyle wierszy ile jest sal lekcyjnych. Jej elementami będzie 1 gdy sala jest wolna oraz 0 gdy jest już zarezerwowana. Tą macierz będziemy oznaczać przez $DS_{s\ c}$.

Kwestię dyspozycyjności klas potraktujemy w ten sam sposób.

Definicja 2.3.
Niech p jest planem typu IV realizyjącym przydział Π. Mówimy że klasa ma konflikt na miejscu i-tym i j-tym gdy $i, j \leqslant \pi, i \neq j$, $\Pi_i = (t_i, d_i, q_i)$, $\Pi_j = (t_j, d_j, q_j)$, $p_i = (c_i, s_i)$, $p_j = (c_j, s_j)$ oraz

$$q_i = q_j \wedge c_i = c_j$$

Także w tym przypadku utworzymy macierz o liczbie wierszy równej liczbie klas oraz liczbie kolumn równej liczbie czasów. Oznaczać ją będziemy $DK_{k\ c}$. Macierz ta domyślnie wypełniona będzie jedynkami, co oznacza, że klasa może mieć przydzieloną lekcję w danym czasie. Również tym przypadku, szukanie konfliktu klas, może odbyć się według metody opisanej powyżej.

Do omówienia pozostaje jeszcze warunek 3.

Definicja 2.4.

Nich c będzie czasem prowadzenia zajęć $c \in C$.

Niech c_d będzie maksymalną liczbą zajęć jakie można przeprowadzić w ciągu jednego dnia.

Niech p będzie planem.

Niech Π jest przydziałem planu p , mający π lekcji.

Okienko klasy q istnieje w czasie o, gdy istnieje $i, j \in 1, \ldots, \pi$ spełniające poniższe warunki:

1. $\Pi_i = (t_i, d_i, q_i)$, $\Pi_j = (t_j, d_j, q_j)$, $p_i = (c_i, s_i)$, $p_j = (c_j, s_j)$
2. $q = q_i \wedge q = q_j$ *- lekcje są prowadzone w klasie q,*
3. $\lceil \frac{c_i}{c_d} \rceil = \lceil \frac{c_j}{c_d} \rceil$ *- lekcje są przeprowadzane w tym samym dniu,*
4. $c_i < o < c_j$ *-czasy przeprowadzenia lekcji różnią się o więcej niż 1.*
5. $\forall_{k \leqslant \pi}((k \neq i) \wedge (k \neq j) \wedge ((t_k, d_k, q_k) \in \Pi) \wedge (t_i = t_k) \Rightarrow (c_k \neq o))$ *nie istnieje lekcja prowadzona przez nauczyciela t_i w czasie o.*

Posiadając matematyczną definicję okienka klasy. Możemy znaleźć narzędzie dzięki któremu będziemy mogli sprawdzić czy warunek numer 3 jest spełniony. Narzędziem tym będzie funkcja licząca okienka klas. Oczywiście aby warunek ten był spełniony będziemy wymagać aby ta funkcja miała dla danego planu wartość 0.

Definicja 2.5. *Niech $p \in PL_4$ jest planem realizującym przydział Π. Funkcją liczby okienek klasy q nazywać będziemy funkcję $\phi_k : PL_4 \times k \to \mathbb{N}$ przyporządkowującą każdemu planowi p i klasie q liczbę okienek klasy q występującą w planie p.*

Definicja 2.6. *Niech $p \in PL_4$ jest planem realizującym przydział Π. Funkcją liczby okienek klas nazywać będziemy funkcję $f_{ok} : PL_4 \to \mathbb{N}$ przyporządkowującą każdemu planowi p liczbę okienek klas występującą w planie p. Funkcje tą można wyrazić wzorem:*

$$f_{ok}(p) = \sum_{q=1}^{k} \phi_k(p, q)$$

Obecnie możemy zdefiniować pojęcie poprawnego planu.

Definicja 2.7.

Niech Π jest π-elementowym przydziałem. Plan p realizujący przydział Π,będziemy nazywać poprawnym planem wtedy i tylko wtedy gdy:

1. *p ma π elementów*

2. *w p nie występują konflikty nauczyciela, sal lub klas.*

3. $f_{ok}(p) = 0$ *- w planie nie ma okienek klas.*

W dalszych rozdziałach, przez plan, będziemy rozumieć plan poprawny. W sytuacji gdy plan będzie niepoprawny będzie to jasno komunikowane.

Rozdział 3

Warunki konieczne istnienia planu lekcji

W tym rozdziale zostaną wyprowadzone warunki konieczne istnienia poprawnego planu lekcji. Wyprowadzone warunki istnienia planu lekcji będą miały charakter mnogościowy i będą wykorzystywane w algorytmach tworzenia planu. Będą one mówiły że planu na pewno nie można ułożyć, ponieważ "czegoś" brakuje: sal, nauczycieli mogących przeprowadzić daną lekcję lub czasu w jakim klasa może ją mieć. Są to warunki konieczne. Ich niespełnienie gwarantuje nam że plan nie istnieje, natomiast spełnienie ich nie gwarantuje istnienia planu.

3.1 Pierwszy warunek konieczny istnienia planu lekcji

Chwilowo załóżmy że każde zajęcia mogą się odbywać w dowolnej sali lekcyjnej. W warunku stosowane będą macierze dyspozycyjności nauczycieli $DN_{n\ c}$ oznaczane dalej przez DN, sal $DS_{s\ c}$oznaczane przez DS oraz klas $DK_{k\ c}$ oznaczane przez DK.

Twierdzenie 3.1. *I Warunek konieczny istnienia planu lekcji*
Niech Π *jest przydziałem lekcji danej szkoły oraz* $\overline{\overline{\Pi}} = \pi$.
Niech N *jest skończonym zbiorem nauczycieli uczących w danej szkole oraz* $\overline{\overline{N}} = n$.
Niech K *jest skończonym zbiorem klas uczonych w danej szkole oraz* $\overline{\overline{K}} = k$.
Niech S *jest skończonym zbiorem sal które posiada dana szkoła oraz* $\overline{\overline{S}} = s$.
Niech C *jest skończonym zbiorem czasów rozpoczęcia lekcji oraz* $\overline{\overline{C}} = c$.
Niech DN *jest macierzą dyspozycji nauczycieli o wymiarach* n *na* c.

Niech DS jest macierzą dostępności sal o wymiarach s na c.

Jeżeli istnieje plan lekcji to $\sum_{j=1}^{c} min(k, \sum_{i=1}^{n} DN_{ij}, \sum_{i=1}^{s} DS_{ij}) \geqslant \pi$.

Dowód.

Załóżmy że plan istnieje. Wtedy w ustalonym czasie c_j może się odbyć maksymalnie $min(k, \sum_{i=1}^{n} DN_{ij}, \sum_{i=1}^{s} DS_{ij})$ ponieważ liczba lekcji, jakie mogą się odbyć w danym czasie, jest mniejsza bądź równa minimum z : liczby klas w szkole równej k, liczby sal dostępnych chwilowo w danym czasie $\sum_{i=1}^{s} DS_{ij}$ oraz liczby nauczycieli dostępnych w danym czasie $\sum_{i=1}^{n} DN_{ij}$. Liczba lekcji w czasie c_j jest oczywiście większa bądź równa 0. Ponieważ liczba wszystkich lekcji jest równa π, przechodząc po wszystkich czasach otrzymujemy żądaną nierówność. □

Na podstawie powyższego warunku oparta jest procedura 1 dająca w odpowiedzi *True* gdy warunek jest spełniony, lub *False* gdy nie może istnieć plan spełniający dany przydział.

Warunek I działa względem kolumn macierzy DN i DS. Sprawą pierwszoplanową w tym warunku jest czas. Następne warunki działają wierszowo na macierzach DN i DS.

3.2 Drugi warunek konieczny istnienia planu lekcji

Warunek ten dotyczy dyspozycyjności nauczycieli.

Twierdzenie 3.2. *II warunek konieczny istnienia planu lekcji*
Niech Π jest przydziałem lekcji danej szkoły oraz $\overline{\overline{\Pi}} = \pi$.
Niech N jest skończonym zbiorem nauczycieli uczących w danej szkole oraz $\overline{\overline{N}} = n$.
Niech K jest skończonym zbiorem klas uczonych w danej szkole oraz $q \in K$.
Niech P jest skończonym zbiorem przedmiotów nauczanych w danej szkole oraz $p \in P$.
Niech $t, t' \in N$.
Niech DN jest macierzą dyspozycji nauczycieli o wymiarach n na c.
Niech l_t jest równa liczbie lekcji $(t', d, q) \in \Pi$ takich że $t' = t$.

Algorithm 1 I warunek konieczny istnienia planu lekcji

Require: π, DN, DS, k, c

Ensure: $True$, $False$

```
s = 0
for j = 1 → c do
    s_DN ← 0
    s_DS ← 0
    for i = 1 → n do
        s_DN ← s_DN + DN_ij
    end for
    for i = 1 → s do
        s_DS ← s_DS + DS_ij
    end for
    s ← s + min(k, s_DN, s_DS)
end for
if s ⩾ π then
    return TRUE
else
    return FALSE
end if
```

Niech $c_t = \sum_{i=1}^{c} DN_{ti}$ *jest równa liczbie czasów w jakich nauczyciel* t *może przeprowadzić lekcje.*

Jeżeli plan lekcji istnieje to:

$$\forall_{t \in N} l_t \leqslant c_t$$

Dowód.

Warunek wynika z prostego faktu. Mianowicie liczba godzin lekcyjnych w jakich nauczyciel może pracować musi być większa lub równa liczbie lekcji jakie ma do przeprowadzenia. □

Na podstawie danego twierdzenia można bardzo łatwo stworzyć procedurę sprawdzającą czy jest spełniony ten warunek.

Algorithm 2 II warunek konieczny istnienia planu lekcji

Require: Π, DN,c, n
Ensure: $True$, $False$
```
for i = 1 → n do
    d ← 0
    for j = 1 → c do
        d = d + DN_ij
    end for
    l ← 0
    for all (t, p, k) ∈ Π do
        if t = i then
            l++
        end if
    end for
    if l > d then
        return False
    end if
end for
if i = n + 1 then
    return TRUE
end if
```

3.3 Trzeci warunek konieczny istnienia planu lekcji

Ten warunek dotyczy klas. Klasie nie można przydzielić więcej lekcji niż jest dostępnych czasów w których takie lekcje można przeprowadzić. Dany warunek można wyrazić następująco:

Twierdzenie 3.3. *III Warunek konieczny istnienia planu lekcji*
Niech Π *jest przydziałem lekcji danej szkoły oraz* $\overline{\overline{\Pi}} = \pi$.
Niech K *jest skończonym zbiorem klas uczonych w danej szkole oraz* $q \in K$.
Niech C *jest skończonym zbiorem czasów rozpoczęcia lekcji oraz* $\overline{\overline{C}} = c$.
Niech $q, q' \in K$.
Niech $(t', d, q) \in \Pi$
Niech l_q *jest równa liczbie lekcji* $(t, d, q') \in \Pi$ *takich że* $q' = q$.

Niech $c_q = \sum_{i=1}^{c} DK_{qi}$ *jest równa liczbie czasów w jakich klasa q może mieć przydzieloną lekcję.*
Plan lekcji istnieje wtedy gdy

$$\forall_{q \in K} l_q \leqslant c_q$$

Dowód.
Dowód powyższego warunku jest trywialny. Oczywiście, gdy istnieje klasa która w danym przydziale ma więcej lekcji niż czasów w jakich te lekcje można przeprowadzić, to nie może istnieć plan lekcji który taki przydział realizuje. Jest to równoważne stwierdzeniu, że jeżeli istnieje plan lekcji to liczba lekcji danej klasy jest mniejsza bądź równa liczbie czasów w jakich ta klasa jest dyspozycyjna. □

Algorithm 3 Sprawdza III warunek konieczny istnienia planu lekcji

```
Require: Π,DK ,c,k
Ensure: True, False
  for i = 1 → k do
      d ← 0
      for j = 1 → c do
          d ← d + DK_ij
      end for
      l_q ← 0
      for all (t, d, q) ∈ Π do
          if q = i then
              l_q + +
          end if
      end for
      if l_q > d then
          return False
      end if
  end for
  if i = c + 1 then
      return TRUE
  end if
```

3.4 Warunek trywialne

Dotychczas sprawdziliśmy tabelę dyspozycyjności nauczycieli, sal oraz klas kolumnowo w warunku I. Warunek II sprawdzał wierszowo tabelę dyspozycyjności nauczycieli z kolei warunek III sprawdzał wierszowo tabelę dyspozycyjności klas. Natomiast nie została sprawdzona wierszowo tabela dyspozycyjności sal. Nie ma to większego uzasadnienia, ponieważ w przeciwności do nauczycieli i klas, sale są zamienne. Dana lekcja może odbywać się bowiem we wszystkich salach. Dzieje się tak ponieważ założyliśmy że każde zajęcia można przeprowadzić w każdej sali. Jedyny warunek jaki może dotyczyć sal jest więc bardzo ogólny. Mianowicie:

Twierdzenie 3.4.
Niech Π *jest przydziałem lekcji danej szkoły oraz* $\overline{\overline{\Pi}} = \pi$.
Niech S *jest skończonym zbiorem sal które posiada dana szkoła oraz* $\overline{\overline{S}} = s$.
Niech DS *jest macierzą dostępności sal o wymiarach* s *na* c.
Niech d_s *oznacza liczbę jedynek w macierzy* DS.
Plan lekcji istnieje gdy $\pi \leqslant d_s$

Dowód.
Dowód wynika z oczywistego faktu iż Każdej lekcji należy przydzielić salę. Więc liczba lekcji w przydziale musi być mniejsza bądź równa liczbie wolnych sal. □

Na podstawie powyższego twierdzenia można napisać bardzo prostą procedurę sprawdzającą ten warunek.

Algorithm 4 Warunek trywialny istnienia planu lekcji

Require: π,DS ,c,s

Ensure: $True$, $False$

```
d_s ← 0
for i = 1 → s do
    for j = 1 → c do
        d_s ← d + DS_ij
    end for
end for
if π ⩽ d_s then
    return True
else
    return False
end if
```

Rozdział 4

Algorytmy tworzenia planu lekcji

W tym rozdziale zostanie wprowadzonych kilka algorytmów tworzenia planu lekcji. Większość z nich będzie algorytmami heurystycznymi, czyli algorytmami dla których nie można udowodnić poprawności działania oraz nie zawsze będą dawać poprawne rozwiązanie. Niektóre algorytmy będą natomiast algorytmami losowymi. Pierwszym algorytmem jaki będzie prezentowany, będzie algorytm możliwie najprostszy, aczkolwiek najmniej wydajny. Następne algorytmy będą modyfikacjami algorytmów poprzednich.

4.1 Algorytmy naiwne

4.1.1 Pierwszy algorytm naiwny

Idea algorytmu jest następująca: mając dany przydział oraz tabele dyspozycyjności nauczycieli i tabelę dyspozycyjności sal sprawdzamy warunek konieczny istnienia planu lekcji. Jeżeli jest on spełniony to próbujemy wstawić w plan pierwszą lekcję z przydziału w pierwszy możliwy termin, to znaczy w pierwszym czasie w jakim mamy do dyspozycji nauczyciela oraz wolną salę. Jeżeli nie może wstawić pierwszej lekcji, to próbuje wstawić drugą itd. Następnie działamy rekurencyjnie do wyczerpania przydziału.

Dany algorytm należy zakończyć gdy przydział będzie zbiorem pustym, co oznaczać będzie, że algorytm zakończył się sukcesem i ułożył plan w którym są wszystkie lekcje z przydziału. Algorytm ten w wyniku swojego działania da "pierwszy z brzegu" plan.

Jeżeli jednak po wstawieniu do planu danej lekcji l nie można wykonać kroku rekurencyjnego, czyli po wstawieniu lekcji l nie można wstawić żadnej innej lekcji z przydziału, to niestety algorytm kończy się porażką.

Teoretyczny czas działania algorytmu, dla przydziału o n lekcjach, wynosi $n!$. W najgorszym przypadku pierwszy krok algorytmu należy powtórzyć dla każdej lekcji z przydziału, czyli n razy, drugi natomiast $n-1$ itd. Sytuacja taka będzie miała miejsce jedynie wtedy gdy w każdym kroku, będzie można dodać do planu tylko ostatnią lekcję.

W sytuacji kiedy planów spełniających przydział będzie dużo, a taka sytuacja ma miejsce najczęściej, algorytm nie będzie musiał wykonywać wielu prób dodania lekcji a jego czas działania będzie wtedy niewiele większy niż n.

W przypadku jaki najczęściej będzie miał miejsce. Czyli w sytuacji kiedy planów spełniających przydział będzie dużo, algorytm nie będzie musiał wykonywać wielu prób dodania lekcji. Wtedy jego czas będzie niewiele większy niż n.

Niech l będzie lekcją z przydziału Π, $DN_{n\ c}$ macierzą dyspozycyjności nauczycieli, $DS_{s\ c}$ będzie tablicą dyspozycyjności sal, $DK_{k\ c}$ będzie tabelą dyspozycyjności klas, p będzie planem. Poniżej przedstawiony zostanie kod procedury 5 dodawania jednej lekcji do planu.

Następnie można utworzyć procedurę rekurencyjną 6 kończącą układanie rozpoczętego już planu zgodnie z przydziałem P oraz z uwzględnieniem macierzy DN, DS i DK.

Ostatnim krokiem będzie rozpoczęcie pracy powyższej procedury od pustego planu, wyjściowego przydziału P i wyjściowych macierzy DN i DS oraz macierzy DK wypełnionej tylko jedynkami. Co pozwoli otrzymać nam procedurę numer 7.

4.1.2 Drugi algorytm naiwny

Następny algorytm powstaje poprzez niewielką modyfikację algorytmu numer 7, a właściwie to przez modyfikację procedury 6 będącej jego podstawą. Modyfikacja polega na dodawaniu lekcji prowadzonej przez najmniej obecnie dyspozycyjnego nauczyciela do planu. Owa dyspozycyjność będzie ilorazem lekcji jakie nauczyciel ma przeprowadzić, czyli takich lekcji w przydziale Π do których jest przypisany dany nauczyciel, oraz czasów w jakich może prowadzić lekcje. Liczbę lekcji do przeprowadzenia oznaczać będziemy przez l_n natomiast liczbę czasów w jakich można dodać lekcję do planu, przez c_n. Dyspozycyjność danego nauczyciela będziemy oznaczać przez d_n.

Jeżeli nauczyciel występuje w przydziale Π to oznacza że $l_n \geqslant 1$.A aby plan mógł

Algorithm 5 Dodaj

Require: $l = (t, d, q)$, DN, DS DK, p, s

Ensure: p, DN, DS, DK

```
i ← 1
j ← 1
while i ⩽ s do
    while j ⩽ c do
        if DN_{t,j} = 1 ∧ DS_{i,j} = 1 ∧ DK_{q,j} = 1 then
            p ← p + (j, i)                    ▷ Dodanie lekcji w czasie j w sali i .
            DN_{t,j} ← 0
            DS_{i,j} ← 0
            DK_{q,j} ← 0
            return p, DN, DS, DK
        end if
        j + +
    end while
    i + +
end while
Pisz "Nie można dodać lekcji l"
return p, DN, DS, DK
```

Algorithm 6 Generuj plan rekurencyjnie

Require: Π, π p, DN, DS, DK
Ensure: p

```
i ← 1
while i ⩽ π do
    l ← Π[i]
    if Dodaj(l, DN, DS, DK, p, s)= p then    ▷ Warunek zachodzi gdy nie udało się dodać lekcji l do planu p, czyli gdy funkcja Dodaj nie zmodyfikowała planu.
        i++
    else
        Π ← Π\{l}
        if Warunek Konieczny Istnienia Planu = True then
            p ← Generuj plan rekurencyjnie(Π,π − 1, p, DN, DS, DK)
            return p
        else
            Pisz "Nie można stworzyć planu"
        end if
    end if
    i + +
end while
if i > długość Π then
    Pisz "Nie można stworzyć planu"
end if
```

Algorithm 7 Generuj plan

Require: DN, DS, Π, π
Ensure: p

```
p ←pusty plan
DK ←macierz wypełniona jedynkami
if Warunek Konieczny Istnienia Planu = True then
    p ← Generuj plan rekurencyjnie(Π, π, p, DN, DS, DK)
else
    Pisz "Nie można stworzyć planu"
end if
```

istnieć to $c_n \geqslant l_n$. Stąd

$$0 < \frac{l_n}{c_n} \leqslant 1$$

Więc $d_n \in (0, 1]$. Współczynnik d_n będzie się zwiększał przy wzroście liczby lekcji oraz przy zmniejszeniu liczby czasów, co oznacza że będzie duży dla nauczyciela który ma wiele godzin lekcyjnych oraz mały dla nauczyciela który ma niewielką ilość lekcji które może przeprowadzić w wielu czasach. Idea algorytmu polega na dodawaniu w pierwszej kolejności lekcji nauczyciela najbardziej zapracowanego, czyli takiego którego współczynnik d_n jest duży

Algorithm 8 Generuj plan na podstawie dyspozycji

Require: Π, p DN, DS, DK, n, c, s

Ensure: p

```
p ←pusty plan
d[] ← pusta tablica
for i = 1 → n do                         ▷ Przygotowanie tablicy dyspozycji
    d[i] ← LiczbaWolnychLekcji(i) / LiczbaWolnychCzasow(i)
end for
while Π ≠ ∅ do
    m ← MAX(d[])                         ▷ Poszukiwanie najmniej dyspozycyjnego nauczyciela
    for i = 1 → n do
        if d[] = m then
            b ← i
        end if
    end for
    j ← 1
    (n′, p, k) ← Π[j]
    sukces ← False
    while j ⩽ n do
        if n′ = b then
            if Dodaj((n′, p, k), DN, DS, DK, p, s)≠ p then
                j ← s
                sukces ← True
            end if
            j + +
        end if
    end while
    if sukces = True then
        d[b] ← (LiczbaWolnychLekcji(i)−1) / (LiczbaWolnychCzasow(i)−1)
    else
        Pisz ”Błąd-nie można stworzyć planu.”
    end if
end while
```

Rozdział 5

Wyprowadzenie funkcji celu

W poprzednich rozdziałach przedstawione zostały modele oraz algorytmy tworzenia planu lekcji. Obecnie potrzebujemy funkcji przyporządkowującej wartości planom lekcji. Chcemy aby funkcja ta zależała od kilku elementów.

Szkoła, zazwyczaj, jest organizacją która nie jest nastawiona na generowanie zysku. Nie możemy mówić bezpośrednio o planie optymalnym ze względu na dochód. Natomiast możemy zoptymalizować koszt wykonania przydziału, czyli postarać się zredukować koszty działalności szkoły. Zazwyczaj największym kosztem, na który można wpłynąć, jest utrzymanie (a w szczególności ogrzanie) sal lekcyjnych. W związku z tym dążymy do planu realizowanego w jak najmniejszej liczbie sal. W planie chcemy też zadbać o komfort i higienę pracy nauczycieli oraz uczniów. W tym celu szukamy planu który miałby jak najmniejszą liczbę "okienek", patrząc z perspektywy zarówno nauczycieli jak i klas. ("Okienko" to sytuacja w której nauczyciel lub klasa ma lukę w czasie pracy np. ma lekcję 1 i 3, ale nie ma lekcji 2.) Ważne jest też, aby w miarę możliwości, w planie nie występowało zbyt wiele zmian sal. Sama zmiana sali przez klasę lub nauczyciela nic nie kosztuje. Jednak wiąże się ona z przekazaniem klucza do sali, przenoszeniem pomocy dydaktycznych, podłączeniem laptopów do rzutników multimedialnych itp. Czynności te są czasochłonne oraz losowo generują problemy opóźniające lub nawet uniemożliwiające prowadzenie lekcji. Dlatego dążymy do jak najmniejszej liczby zmienionych sal przez nauczyciela oraz przez klasę. Nasza funkcja musi być więc zależna od ilości zmian sal. Chcemy też aby plan, lepszy pod jakimś względem, miał niższą wartość funkcji, wtedy optymalizacja danej funkcji będzie polegała na szukaniu minimum tej funkcji. Podsumowując: poszukujemy funkcji wprost proporcjonalnej względem kosztów wykonania planu, liczby okienek nauczyciela oraz liczby zmian sal przez nauczyciela. Dodatkowo zakładamy, że poszukujemy planu bez okienek klas.

Oczywiście istnieje nieskończenie wiele funkcji spełniających powyższe postulaty. Za funkcją, która zostanie obecnie przedstawiona, przemawia jedynie intuicja.

5.1 Pierwsza propozycja funkcji celu

Zdefiniujmy składowe naszej funkcji celu. Na początek zdefiniujmy zależność planu i sal.

Definicja 5.1. *Niech $p \in PL_4$ jest planem typu IV realizującym przydział Π. Funkcją liczby sal nazywać będziemy funkcję $f_s : PL_4 \to \mathbb{N}$ przyporządkowującą każdemu planowi p liczbę sal potrzebną do jego wykonania.*

Następnym elementem naszej funkcji celu jest zależność planu od liczby okienek. Obecnie należy zdefiniować pojęcie okienka, oraz funkcji przypisującej każdemu planowi liczbę okienek jakie w tym planie, występują. Należy to zrobić dla okienka które ma nauczyciel, i oddzielnie dla okienka klasy.

Najpierw zajmijmy się okienkami nauczycieli.

Definicja 5.2.
Nich c będzie czasem prowadzenia zajęć $c \in C$.
Niech c_d będzie maksymalną liczbą zajęć jakie można przeprowadzić w ciągu jednego dnia.
Niech p będzie planem.
Niech Π jest przydziałem planu p , mający π lekcji.
Okienko nauczyciela t istnieje w czasie o, gdy istnieje $i, j \in 1, \ldots, \pi$ spełniające poniższe warunki:

1. $\Pi_i = (t_i, d_i, q_i)$, $\Pi_j = (t_j, d_j, q_j)$, $p_i = (c_i, s_i)$, $p_j = (c_j, s_j)$
2. $t = t_i \wedge t = t_j$ *- lekcje są prowadzone przez nauczyciela t,*
3. $\lceil \frac{c_i}{c_d} \rceil = \lceil \frac{c_j}{c_d} \rceil$ *- lekcje są przeprowadzane w tym samym dniu,*
4. $c_i < o < c_j$ *-czasy przeprowadzenia lekcji różnią się o więcej niż 1.*
5. $\forall_{k \leqslant \pi}((k \neq i) \wedge (k \neq j) \wedge ((t_k, d_k, q_k) \in \Pi) \wedge (t_i = t_k) \Rightarrow (c_k \neq o))$ *nie istnieje lekcja prowadzona przez nauczyciela t_i w czasie o.*

Definicja 5.3. *Niech $p \in PL_4$ jest planem realizującym przydział Π. Funkcją liczby okienek nauczyciela t nazywać będziemy funkcję $\phi_n : PL_4 \times n \to \mathbb{N}$ przyporządkowującą każdemu planowi p i nauczycielowi t liczbę okienek nauczyciela t występującą w planie p.*

Definicja 5.4. *Niech $p \in PL_4$ jest planem realizującym przydział Π. Funkcją liczby okienek nazywać będziemy funkcję $f_{on} : PL_4 \to \mathbb{N}$ przyporządkowującą każdemu planowi p liczbę okienek nauczycieli występującą w planie p. Funkcje tą można wyrazić wzorem:*

$$f_{on}(p) = \sum_{t=1}^{n} \phi_n(p, t)$$

Ostatnim elementem naszej funkcji celu będzie zależność planu od liczby zmian. Obecnie zdefiniujemy zmianę sali i funkcję zmiany sali przez nauczyciela.

Definicja 5.5.

Nich c będzie czasem prowadzenia zajęć $c \in C$.

Niech c_d będzie maksymalną liczbą zajęć jakie można przeprowadzić w ciągu jednego dnia.

Niech p będzie planem.

Niech Π jest przydziałem planu p , mający π lekcji.

Zmiana z istnieje gdy istnieje $i, j \in 1, \ldots, \pi$ spełniające poniższe warunki:

1. $\Pi_i = (t_i, d_i, q_i)$, $\Pi_j = (t_j, d_j, q_j)$, $p_i = (c_i, s_i)$, $p_j = (c_j, s_j)$
2. $t_i = t_j$ *-nauczyciel dla obu lekcji jest identyczny*
3. $\lceil \frac{c_i}{c_d} \rceil = \lceil \frac{c_j}{c_d} \rceil$ *- lekcje są przeprowadzane w tym samym dniu*
4. $c_i + 1 = c_j$ *- czasy prowadzenia lekcji różnią się o 1.*
5. $s_i \neq s_j$ *- sale dla obydwóch lekcji są różne.*

Definicja 5.6. *Niech $p \in PL_4$ jest planem realizującym przydział Π. Funkcją liczby zmian nazywać będziemy funkcję $f_z : PL_4 \to \mathbb{N}$ przyporządkowującą każdemu planowi p liczbę zmian sal w nim występującą.*

Obecnie mamy zdefiniowane wszystkie elementy potrzebne do zaproponowania funkcji celu.

Definicja 5.7. *Funkcją celu planu $p \in PL_4$ realizującego przydział Π, zależną od kosztu utrzymania sali, kosztów okienka i kosztów zmiany $k_s, k_o, k_z \in \mathbb{Q}_+$ nazywać będziemy funkcję $f : PL_4 \to \mathbb{Q}_+$ określoną w następujący sposób:*

$$f(p) = k_s f_s(p) + k_o f_{on}(p) + k_z f_z(p)$$

Na początek zajmiemy się funkcją kosztu. Na koszt funkcjonowania szkoły wpływ mają pensje nauczycieli, pracowników administracyjnych oraz koszt utrzymania sal. Jedynie ta ostatnia pozycja zależy od planu lekcji. Salę, w której odbyła się chociaż jedna lekcja w ciągu dnia należy posprzątać oraz ogrzać co generuje określone koszty. Procedurę liczącą liczbę używanych sal - czyli wartość $f_s(p)$ przedstawia procedura 9.

Algorithm 9 f_s

Require: p, c_d, c, s ▷ c_d-maksymalna liczba odbywających się dziennie zajęć.
Ensure: *koszt*
  $S[s, \lceil\frac{c}{c_d}\rceil] \leftarrow$ tablica wypełniona zerami
  for all $(s', c') \in p$ **do**
    $S[s', \lceil\frac{c'}{c_d}\rceil] \leftarrow 1$
  end for ▷ w $S[i,j]$ jest 1, gdy sala i jest chociaż raz używana w dniu j
  for $i = 1 \rightarrow s$ **do**
    for $j = 1 \rightarrow \lceil\frac{c}{c_d}\rceil$ **do**
      $koszt \leftarrow koszt + S[i,j]$
    end for
  end for
  return *koszt*

Teraz należy zająć się funkcją okienek nauczyciela. Najłatwiej jest policzyć okienka każdego z nauczycieli z osobna, a na końcu zsumować wyniki. Aby policzyć okienka danego nauczyciela należy jego lekcje uporządkować względem czasu. Następnie sprawdzamy dla każdej lekcji l_i z osobna, czy dana lekcja oraz lekcja następna l_{i+1} są przeprowadzane tego samego dnia. Jeżeli tak to liczbę okienek można zwiększyć o $c_{i+1} - (c_i + 1)$. W przypadku gdy lekcje następują po sobie dane wyrażenie ma wartość 0, natomiast w innym przypadku liczy liczbę powstałych okienek. Algorytm ten przedstawi procedura 10.

Do omówienia pozostała funkcja zmian. Tak jak funkcję okienek łatwiej jest ją rozbić na zmiany każdego nauczyciela a następnie zsumować wyniki. Sama zmiana sali jest prostsza w opisie matematycznym niż okienko. Wystarczy znaleźć dwie lekcje tego samego nauczyciela przeprowadzane w sąsiednim czasie tego samego dnia w różnych salach. Wykonywać to będzie procedura 11.

Zmiana planu w jednym miejscu (zmiana przydzielonej sali lub przydzielonego czasu lekcji) może spowodować stworzenie planu z błędami.

Algorithm 10 f_{on}

Require: p, Π, π c_d, n, $\triangleright$ c_d-maksymalna liczba odbywających się dziennie zajęć.

Ensure: $wynik$

```
wynik ← 0
for i = 1 → n do
    lekcje ←pusta lista
    for j = 1 → π do
        (t, d, q) ← Π[j]
        (c′, s′) ← p[i]
        if t = i then
            lekcje ← lekcje ∪ (c′, s′)
        end if
    end for
    sortuj lekcje niemalejącą względem wartości c′
    j = 1
    while j ⩽ długość listy lekcje -1 do
        (c₁, s₁) ← lekcje[j]
        (c₁, s₁) ← lekcje[j + 1]
        if ⌈c₁/c_d⌉ = ⌈c₂/c_d⌉ then
            wynik ← wynik + c₂ − c₁ − 1
        end if
    end while
end for
return wynik
```

Algorithm 11 f_z

Require: p, Π, π c_d, n,

Ensure: $wynik$

$wynik \leftarrow 0$
for $i = 1 \rightarrow n$ **do**
  $lekcje \leftarrow$*pusta lista*
  for $j = 1 \rightarrow \pi$ **do**
    $(t, d, q) \leftarrow \Pi[j]$
    $(c', s') \leftarrow p[i]$
    if $t = i$ **then**
      $lekcje \leftarrow lekcje \cup (c', s')$
    end if
  end for
  sortuj $lekcje$ niemalejąco względem wartości c'
  $j = 1$
  while $j \leqslant$ długość listy $lekcje$ -1 **do**
    $(c_1, s_1) \leftarrow lekcje[j]$
    $(c_1, s_1) \leftarrow lekcje[j + 1]$
    if $(\lceil c_1/c_d \rceil = \lceil c_2/c_d \rceil) \wedge (c_i + 1 = c_2) \wedge (s_1 \neq s_2)$ **then**
      $wynik \leftarrow wynik + 1$
    end if
  end while
end for
return $wynik$

5.2 Własność wyprowadzonej funkcji celu

W tym paragrafie postaramy się ograniczyć różnicę wartości funkcji celu pomiędzy dwoma "podobnymi" planami.

Jak można rozumieć podobieństwo planów? Za plany podobne należy uznać plany które różnią się tylko jedną lekcją.

Definicja 5.8.
Plany p i p' realizujące π elementowy przydział Π są podobne gdy:

$$\exists_{j\in\{1,2,\ldots,\pi\}}\forall_{i\in\{1,2,\ldots,\pi\}}(i \neq j \wedge p_i = p'_i) \vee (i = j \wedge p_i \neq p'_i)$$

Dwa podobne plany mogą się różnić tylko salą wykonywania j-tej lekcji, lub jej czasem wykonywania.

Zmiana sali może spowodować powstanie maksymalnie dwóch zmian sal w rozumieniu definicji 5.6. Jeżeli nauczyciel miał trzy kolejne lekcje w tej samej sali, a zmieniono mu salę lekcji drugiej, to powstały w ten sposób dwie nowe zmiany, czyli wartość f_z zwiększyła się o 2. Jest to sytuacja najgorsza z punktu widzenia wartości funkcji f_z.

Zmiana sali wykonywania lekcji może też zwiększyć liczbę utrzymywanych sal, czyli wartość f_s o 1. Dzieje się tak, gdy lekcja, zostanie przeniesiona do sali która w danym dniu nie była używana.

Zmiana czasu pojedynczej lekcji może spowodować te same problemy co zmiana sali. Dodatkowo, dla planu, w którym dziennie może odbywać się c_d lekcji może wygenerować c_d-2 okienek. Dzieje się tak np. gdy nauczycielowi który miał w danym dniu zaplanowaną tylko lekcję pierwszą i drugą, czas lekcji drugiej zostanie zmieniony na ostatni czas w danym dniu.

Jednoczesna zmiana sali i czasu danej lekcji nie zwiększy wartości funkcji celu bardziej niż zmiana tylko czasu lekcji.

Twierdzenie 5.1.
Niech plany p i p' realizują przydział Π i są podobne. Niech c_d jest maksymalną liczbą czasów rozpoczęcia lekcji w dniu, $k_s, k_o, k_z \in \mathbb{Q}_+$ gdzie k_s jest kosztem utrzymania sali, k_o jest kosztem okienka, k_z jest kosztem zmiany. Wtedy:

$$|f_k(p) - f_k(p')| \leqslant k_s + 2k_z + (c_d - 2)k_o$$

Za dowód twierdzenia można uznać trzy powyższe akapity.

Rozdział 6

Optymalizacja planu lekcji

Rozdział ten jest najważniejszą częścią powyższej pracy. W poprzednich rozdziałach zbudowanych zostało kilka modeli matematycznych planu lekcji, przedstawiony został algorytm tworzenia planu oraz wyprowadzono funkcję celu. Co umożliwiło opisanie przestrzeni w jakiej się poruszamy. Umiemy wygenerować punkt startowy, co jest bardzo często pierwszą czynnością do wykonania w wielu metod optymalizacyjnych. Dzięki funkcji celu potrafimy porównać otrzymane wyniki. Dysponujemy obecnie odpowiednimi narzędziami, aby na potrzeby naszego problemu, stworzyć kilka użytecznych algorytmów optymalizacyjnych.

6.1 Algorytm zachłanny

"Algorytmy służące do rozwiązania problemów optymalizacyjnych polegają zwykle na podjęciu decyzji - w każdym kroku należy wybrać jedną z wielu możliwości. ...
Algorytm zachłanny (ang. greedy algorithm) wykonuje zawsze działanie, które wydaje się w danej chwili najkorzystniejsze. Wybieramy zatem lokalnie optymalną możliwość w nadziei, że doprowadzi ona do globalnie optymalnego rozwiązania."
- tak opisane zostały algorytmy zachłanne w [2].

Algorytmy zachłanne polegają na wyborze najlepszej chwilowo ("lokalnie") decyzji, w trakcie budowy rozwiązania, w wyniku jej podjęcia, rozpatrywany problem zmniejsza się do podproblemu o mniejszym rozmiarze. Dzięki czemu, większość algorytmów zachłannych rozwiązujących konkretny problem, jest algorytmami poprawnymi i dodatkowo często dającymi poprawne rozwiązanie. Nie zawsze jednak można udowodnić, że rozwiązanie wyznaczone przez algorytm zachłanny, jest rozwiązaniem optymalnym. Często, tworząc algorytm zachłanny, kierujemy się jedynie heurystyką. Przeczuwamy, że dane postępowanie, da w rezultacie bardzo dobry (a często optymalny) wynik.

Jak może wyglądać algorytm zachłanny próbujący wygenerować optymalny plan lekcji? Może on być pewnym rozwinięciem algorytmu *Generuj plan na podstawie dyspozycji* 8. We wspomnianej procedurze, tworząc plan lekcji, staraliśmy się dodawać lekcję nauczyciela, który jest najmniej dyspozycyjny czyli iloraz liczby lekcji jakie ma do przeprowadzenia przez liczbę godzin w jakich te lekcje może prowadzić, jest maksymalny. Po wybraniu nauczyciela, dodawaliśmy lekcję w pierwszy wolny termin, w pierwszej wolnej sali. Podejście to nie było dobre pod względem optymalizacji. Niewielka modyfikacja procedury *Generuj plan na podstawie dyspozycji* pozwoli uznać ją za procedurę realizującą algorytm zachłanny. Owa modyfikacja polega na wyborze najmniej dyspozycyjnej klasy, najlepszego czasu oraz najlepszej sali, pod względem optymalizacji funkcji kosztu, dla dodawanej lekcji.

Jak należy wybrać najmniej dyspozycyjną klasę? Na podstawie przydziału należy wybrać klasy w jakich nauczyciel ma prowadzić lekcje. Następnie obliczyć trzeba, ile jeszcze jest, w każdej z nich z osobna lekcji do przydzielenia oraz ile mają nieprzydzielonych czasów. Współczynnik dyspozycyjności klas można przedstawić jako iloraz

$$\frac{Liczba\ Lekcji\ Do\ Zaplanowania}{Liczba\ Nieprzydzielonych\ Czasow}$$

Oczywiście wybierzemy klasę z najwyższym współczynnikiem.

Co oznacza dobór najlepszego czasu? Wybrany czas nie powinien tworzyć okienka klasy. Jednak w trakcie działania algorytmu powstałe okienko może zostać zapełnione przez inną lekcję, dzięki czemu utworzony plan może być planem poprawnym. Chcemy też aby czas dodawanej lekcji nie tworzył okienek nauczyciela oraz aby nauczyciel przeprowadzał swoje lekcje w możliwie jak najmniejszej liczbie dni. Osiągniemy to poprzez zabieg analogiczny, do przyporządkowania nauczycielom współczynnika dyspozycyjności. Każdemu z czasów, w których dany nauczyciel jest dyspozycyjny, przyporządkujemy współczynnik według którego będziemy je następnie segregować nierosnąco. Musi on zależeć przede wszystkim od planu klasy. Współczynnik ten może składać się ze składowej odpowiedzialnej za okienka klasy oraz składowej odpowiedzialnej za okienka nauczycieli. Składowa okienek nauczycieli może być obliczana w następujący sposób:

- 0 dla czasów w dni w które nauczyciel nie prowadzi lekcji
- $\frac{1}{2}$ dla czasów w dni w które nauczyciel prowadzi lekcje

W ten sposób faworyzujemy czasy rozpoczęcia lekcji, które są w dni, w których nauczyciel już pracuje. Obecnie, jeżeli nauczyciel pracuje na trzeciej godzinie lekcyjnej, to współczynniki czasów lekcji pierwszej, drugiej, czwartej, a nawet ostatniej w danym dniu, są sobie równe. Aby taka sytuacja nie miała już miejsca każdą składową czasu nauczyciela, z dnia przeprowadzenia zaplanowanej lekcji, można zwiększyć o pewną wartość odwrotnie proporcjonalną do różnicy pomiędzy wspomnianymi czasami. Jeżeli nauczyciel n_p prowadzi już lekcję w czasie c_p to czasowi c_i z tego samego dnia co c_p możemy zwiększyć składową czasu nauczyciela o Np. $(\frac{1}{2})^{|c_p - c_i|}$. Możemy tak postąpić względem każdej z zaplanowanych lekcji. Dzięki temu w sytuacji, w której nauczyciel ma zaplanowaną lekcję drugą oraz czwartą to największą składową czasu nauczyciela będzie miała lekcja trzecia (równa $\frac{1}{2} + (\frac{1}{2})^{|2-3|} + (\frac{1}{2})^{|4-3|} = 1\frac{1}{2}$). Natomiast lekcja pierwsza oraz piąta będą miały przyporządkowane dużo mniejszy współczynnik (równy $\frac{1}{2} + \frac{1}{2}^{|1-2|} + (\frac{1}{2})^{|1-4|} = 1\frac{1}{8}$ oraz $\frac{1}{2} + \frac{1}{2}^{|5-2|} + (\frac{1}{2})^{|5-4|} = 1\frac{1}{8}$). Dzięki takiej budowie składowej czasu nauczyciela, czasy które są w dni, w które nauczyciel ma już kilka lekcji, będą większe od składowej czasów nauczyciela z dni w które nauczyciel ma niewiele lekcji. Zaproponowana budowa współczynnika zapewni "przyciąganie" się zaplanowanych lekcji prowadzonych przez danego nauczyciela.

Analogicznie można obliczyć składową czasu klasy. Mając obydwie składowe współczynnik czasu można zdefiniować jako sumę podwojonej wartości składowej czasu klasy oraz składowej czasu nauczyciela. Dzięki zwiększeniu wartości składowej czasu klasy, okienka klasy będą miały niewielką szansę powstania. tym samym szansa na uzyskanie poprawnego planu będzie duża.

Podobny zabieg zastosowany zostanie też do wyboru sal. Po ustaleniu czasu w jaki chcemy daną lekcję dodać do planu stworzymy listę wolnych sal. Każdej sali przyporządkujemy współczynnik sali. Może on być przyporządkowany następująco:

- 1 dla sal w których w danym dniu są już zaplanowane lekcje
- $\frac{1}{2}$ dla sal w których zaplanowane są lekcje tylko w inne dni.
- 0 dla sal w których jeszcze nie zaplanowano lekcji

W wyniku tego, preferowane w wyborze, będą sale dla których już musimy ponieść koszt utrzymania.

Następnie aby nie wymuszać zmian sali, jeżeli staramy się dodać lekcję w czasie c_p oraz istnieje lekcje w czasie $c_p - 1$ lub $c_p + 1$ prowadzone przez obecnego nauczyciela, to zwiększymy współczynnik sal w których te lekcje były prowadzone o np $\frac{1}{2}$.

Podsumowując: w trakcie dodawania lekcji postaramy się wybrać najbardziej zapracowanego nauczyciela, następnie czas w jakim najlepiej jest przeprowadzić lekcję a na końcu odpowiednią salę. Często koszt utrzymania sali jest większy od kosztu jednego okienka, co by przemawiało za zmianą kolejności wyboru na : nauczyciel, sala, czas, za poprzednią kolejnością przemawia natomiast fakt, że źle dobrany czas może stworzyć wiele okienek.

Powyższy algorytm można przedstawić w postaci procedury *Algorytm zachłanny* 12.

6.2 Losowe algorytmy optymalizacyjne

W przypadkach gdy optymalizowana funkcja jest znana. Znamy takie jej własności jak ciągłość i ograniczoność oraz wiele innych cech, można zastosować szereg metod deterministycznych, które przy niewielkim nakładzie obliczeń znajdą ekstremum zadanej funkcji. W sytuacji gdy funkcja celu jest słabo znana i nie jesteśmy w stanie powiedzieć czy ma ona wspomniane już własności, lub pomimo posiadania owych własności, znane algorytmy mogą "utknąć" w jednym z wielu ekstremów lokalnych. W przypadku planu lekcji, gdy przestrzeń w której pracujemy jest wielowymiarową (kilkuset-wymiarową) przestrzenią dyskretną, w której zmiana wartości jednej z wielu zmiennych, może bardzo mocno zmienić wartość funkcji celu (w naszym przypadku funkcji kosztu f_k), często jedynymi skutecznymi metodami okazują się metody niedeterministyczne. A więc algorytmy w których wykorzystana jest losowość.

Losowe algorytmy optymalizacyjne można podzielić na kilka grup. Są to:

- algorytmy typu Monte Carlo,
- algorytmy genetyczne,
- algorytmy ewolucyjne

Przytoczony podział nie jest podziałem ani pełnym, w którym uwzględniono wszystkie rodzaje losowych algorytmów optymalizacyjnych, ani jednoznacznym. Wiele algorytmów może być "przydzielona" do kilku kategorii.

6.2.1 Algorytm optymalizacyjny typu Monte Carlo

"Metodami *Monte Carlo* określa się wszelkie metody wykorzystujące czynnik losowy w rozwiązaniu złożonych problemów, których rozwiązanie metodami analitycznymi

Algorithm 12 Algorytm zachłanny

Require: Π, π DN, DS, n, c, s

Ensure: p

  $p \leftarrow$pusty plan

  $DK[k, c] \leftarrow$ *pusta tablica*

  $dn[n, 2] \leftarrow$ pusta tablica

  for $i = 1 \rightarrow n$ **do** ▷ Przygotowanie tablicy dyspozycji nauczycieli

    $dn[i, 1] \leftarrow i$

    $dn[i, 2] \leftarrow \frac{Liczba\ Wolnych\ Lekcji(i)}{Liczba\ Wolnych\ Czasow(i)}$

  end for

  $dn[,]$ sortuj nierosnąco względem $dn[, 2]$

  $dk[k, 3] \leftarrow$ tablica wypełniona zerami

  for $i = 1 \rightarrow k$ **do** ▷ Przygotowanie tablicy dyspozycji klas

    $dk[i, 1] \leftarrow i$ ▷ Numer klasy

    $dk[i, 2] \leftarrow 1$ ▷ Współczynnik dyspozycyjności i-tej klasy

    for all $(n', k', p') \in \Pi$ **do** ▷ Zliczenie lekcji danej klasy

      if $k' = i$ **then**

        $dk[i, 3] + +$

      end if

    end for

  end for

  for $i = 1 \rightarrow n$ **do**

    $l_i \leftarrow$ *pusta lista lekcji*

    for $j = 1 \rightarrow \pi$ **do** ▷ Szukanie lekcji wybranego nauczyciela

      $(n', p', k') \leftarrow \Pi[j]$

      if $n' = i$ **then**

        $l_i \leftarrow l_i \cup (n', p', k')$

      end if

    end for

    while $l_i \neq \emptyset$ **do**

      $(n_a, p_a, k_a) \leftarrow l_i[1]$

      for all $(n_k, p_k, k_k) \in l_i$ **do** ▷ Szukanie najmniej dyspozycyjnej klasy

        if $dk[k_p, 2] > dk[k_a, 2]$ **then**

          $(n_a, p_a, k_a) \leftarrow (n_k, p_k, k_k)$

        end if

      end for

```
        DC[c, 2] ← tablica zer
        for j = 1 → c do
            DC[j, 1] ← j
            DC[j, 2] ← Wspoczynnik Czasu(i, j, k_a)
        end for
        DC[, ] sortuj nierosnąco względem dn[, 2]
        j = 1
        c_d ← DC[j, 1]
        while (DN[i, d_c] = 1 ∧ DK[k_a, d_c] = 1) do
            j + +
            c_d ← DC[j, 1]
        end while
        ds[s, 2] ← tablica zer
        for j = 1 → s do
            ds[j, 1] ← j
            ds[j, 2] ← Wspoczynnik Sali(j, i, c_d)
        end for
        ds[, ] sortuj nierosnąco względem dn[, 2]
        j ← 1
        while DS[j, c_d] = 0 do
            j + +
        end while
        s_d ← ds[j, 1]
        Dodaj((n_a, p_a, k_a), c_d, s_d)
    end while
end for
```

nie jest możliwe. Metody zaliczane do tej grupy, wykorzystują elementy rachunku prawdopodobieństwa, analizę statystyczną itp. Zdobywają one coraz większą popularność w wielu dziedzinach, w tym również w optymalizacji, bowiem pozwalają podnieść skuteczność działania rozważanych algorytmów przy stosunkowo niewielkim dodatkowym nakładzie pracy.” Powyższy opis zaczerpnięty został z [1]. Wskazuje on na równoznaczność pojęcia algorytmu losowego oraz algorytmu typu *Monte Carlo*. W literaturze można też spotkać sytuację w której algorytmami typu *Monte Carlo* nazywane są jedynie te algorytmy losowe, które zawsze dają jakiś wynik, ale z prawdopodobieństwem większym niż $\frac{1}{2}$ wynik ten jest poprawny.

Algorytmy typu *Monte Carlo* są też definiowane jako randomizowane algorytmy w których czynnik losowy wpływa na jakość wyniku.

Algorytm *Monte Carlo* dla planu lekcji będzie bardzo prostym algorytmem. Jego idea polegać będzie na wyborze, spośród dużej liczby losowo wygenerowanych planów, najlepszego planu spełniającego dany przydział. Powyższy algorytm można zapisać w pseudokodzie w postaci procedury 13.

Algorithm 13 Optymalizacja planu MC

Require: Π, π p, DN, DS, DK, *liczba prob*, p, u, c_d, c, s

Ensure: p

$p \leftarrow Generuj\ plan\ MC(\Pi, \pi, DN, DS, DK)$

$f \leftarrow f_k(p, u, c_d, c, s)$

for $i = 1 \rightarrow liczba\ prob$ **do**

  $k \leftarrow generuj\ losowo\ plan(\Pi, \pi, DN, DS, DK)$

  $fk \leftarrow f_k(p, u, c_d, c, s)$

  if $fk < f$ **then**

    $k \leftarrow fk$

    $p \leftarrow k$

  end if

end for

return p

Aby powyższy algorytm działał, niezbędne jest stworzenie procedury *generuj plan MC*. W poprzednich rozdziałach wyprowadzone zostały algorytmy generujące plan. Jednak każdy z nich był algorytmem deterministycznym, nienadającym się one do zastąpienia procedury *generuj plan MC*. Po ich niewielkiej modyfikacji, można jednak napisać tą procedurę. Przedstawione teraz zostaną procedury oparte na procedurach: *Dodaj* 5, *Generuj plan rekurencyjnie* 6 oraz oczywiście *Generuj plan* 7 które zostały tak zmodyfikowane aby wykorzystywać czynnik losowy.

Największej modyfikacji ulegnie procedura *Dodaj*. Procedura *Dodaj MC*, na niej oparta, zamiast dodawać lekcję w "pierwszy z brzegu" wolny termin w którym możliwe jest jej przeprowadzenie, będzie losować czas z czasów w jakich jest jednocześnie dyspozycyjny nauczyciel i klasa, a następnie losować wolną salę do przeprowadzenia lekcji w danym czasie. Oczywiście może się zdarzyć, że nie będzie istniał żaden czas w którym jednocześnie jest dyspozycyjny nauczyciel i klasa. W takiej sytuacji danej lekcji nie można dodać. Może się zdarzyć też, że można wylosować odpowiedni czas, natomiast nie można w tym czasie znaleźć wolnej sali. Wówczas procedura spróbuje wylosować inny czas. Wynik tych modyfikacji przedstawia procedura 14.

Natomiast procedury *Generuj plan rekurencyjnie* 6 i *Generuj plan rekurencyjnie MC* 15, oraz *Generuj plan* 7 i *Generuj plan MC* 16 różnią się pomiędzy sobą tylko kosmetyczną zmianą w jednej linii kodu.

6.2.2 Genetyczny algorytm optymalizacyjny

Algorytmy genetyczne polegają na wylosowaniu pewnej populacji danych, a następnie "przeprowadzeniu ewolucji". Algorytm buduje kilka "pokoleń" danych w taki sposób, że największe prawdopodobieństwo przeżycia i "rozmnożenia się" mają dane które są dla nas najlepsze. Owe dane zachowują się jak prawdziwe geny w przyrodzie; geny które wpływają na lepsze przystosowanie osobnika do życia są częściej przekazywane następnemu pokoleniu. Natomiast geny które powodują choroby oraz osłabiają organizm giną wraz z nim. gen jednak nie stanowi całego organizmu, lecz tylko jego część. Cały organizm-pakiet genów nazywamy chromosomem. Tak samo jak w przyrodzie w algorytmie będzie dochodzić do krzyżowania oraz mutacji chromosomów. Kiedy należy zakończyć algorytm? Możemy go skończyć po stworzeniu określonej liczby pokoleń lub w momencie gdy przez kilka pokoleń najlepszy wynik się nie poprawił.

Czas na bardziej szczegółowy opis metody. Przez chromosom rozumiemy reprezentanta obiektu, a przez gen najbardziej elementarną część chromosomu. Np. chromosomem może być liczba a genem jeden jej bit. Populacją nazywamy zbiór chromosomów, zaś funkcją przystosowania nazywamy funkcję działającą na genach - jest to zazwyczaj funkcja optymalizacyjna. Algorytm genetyczny może działać według schematu 17:

Przeanalizujmy kroki algorytmu.

Inicjalizacja jest to faza w której losujemy ustaloną ilość chromosomów. Liczba

Algorithm 14 Dodaj MC

Require: $l = (t, d, q)$, DN, DS, DK, p, s
Ensure: p, DN, DS, DK

```
Lista Czasow ← pusta lista
for i = 0 → c do
    if DN_{t,j} = 1 ∧ DK_{q,j} = 1 then
        dopisz do Lista Czasow i
    end if
end for
while Lista Czasow ≠ ∅ do
    j ← losowy element z Lista Czasow
    usuń j z Lista Czasow
    Lista Sal ← pusta lista
    for k = 1 → s do
        if DS_{k,j} = 1 then
            dopisz do Lista Sal k
        end if
    end for
    if Lista Sal ≠ ∅ then
        l ← losowy element z Lista Sal
        p ← p + (j, k)                    ▷ Dodanie lekcji w czasie j w sali k.
        DN_{t,j} ← 0
        DS_{l,j} ← 0
        DK_{q,j} ← 0
        return p, DN, DS, DK
    end if
end while
Pisz "Nie można dodać lekcji l"
return p, DN, DS, DK
```

Algorithm 15 Generuj plan rekurencyjnie MC

Require: Π, π p, DN, DS, DK

Ensure: p

$i \leftarrow 1$

while $i \leqslant \pi$ **do**

$l \leftarrow \Pi[i]$

if Dodaj MC(l, DN, DS, DK, p, s)= p **then** $\triangleright$ Warunek zachodzi gdy nie udało się dodać lekcji l do planu p, czyli gdy funkcja **Dodaj MC** nie zmodyfikowała planu.

i++

else

$\Pi \leftarrow \Pi \backslash \{l\}$

if Warunek Konieczny Istnienia Planu = $True$ **then**

$p \leftarrow$ Generuj plan rekurencyjnie(Π,$\pi - 1$, p, DN, DS, DK)

return p

else

Pisz "Nie można stworzyć planu"

end if

end if

$i++$

end while

if $i >$ długość Π **then**

Pisz "Nie można stworzyć planu"

end if

Algorithm 16 Generuj plan MC

Require: DN, DS, Π, π

Ensure: p

$p \leftarrow$pusty plan

$DK \leftarrow macierz\ wypeniona\ jedynkami$

if Warunek Konieczny Istnienia Planu = $True$ **then**

$p \leftarrow$ Generuj plan rekurencyjnie MC(Π, π, p, DN, DS, DK)

else

Pisz "Nie można stworzyć planu"

end if

Algorithm 17 Algorytm genetyczny

Require: *Liczba prob*

Ensure: *Best*

Inicjacja populacji

$Liczba\ krokow = 0$

for $k = 1 \rightarrow Liczba\ prob$ **do**

Elitaryzm

Selekcja

Krzyżowanie

Mutacja

end for

$Best \leftarrow najlepszy\ chromosom\ z\ populacji$

return *Best*

chromosomów nie może być zbyt mała, ponieważ wtedy metoda nie zadziała dobrze. Mała pula chromosomów nie sprzyja bowiem ewolucji. Natomiast jeżeli liczba ta będzie za duża to algorytm będzie działał bardzo wolno, a efekty jego działania nie będą dużo lepsze niż przy dobrze dobranej liczbie chromosomów. Niestety nie ma dobrej odpowiedzi na pytanie jaka jest dobra liczba chromosomów w danym problemie. Liczba ta też musi być zależna od wartości błędu metody na jaki się godzimy (czyli na wartość różnicy wartości faktycznego minimum i minimum osiągniętego przez metodę).

Następne kroki algorytmy, które są wykonywane w pętli, są krokami tworzenia nowej populacji. Pierwszym krokiem pętli jest **elitaryzm**. Krok ten polega na skopiowaniu ustalonej liczby najlepszych chromosomów do nowej populacji. A więc na "przedłużeniu życia" najlepszych osobników. Nie ma on analogii w przyrodzie, jednak dzięki niemu nie jesteśmy w stanie popsuć już raz osiągniętego wyniku.

Dalej należy dokonać **selekcji**. Jest to proces wyboru pary chromosomów będących parą rodzicielską dla chromosomów nowej populacji. Wyboru należy dokonać w taki sposób, aby chromosomy lepiej przystosowane miały większe prawdopodobieństwo wylosowania od chromosomów słabo przystosowanych. Można to zrobić przy pomocy metody ruletki, polegającej na przyporządkowaniu każdemu chromosomowi przedziału o długości proporcjonalnej do szansy jego wylosowania. Wtedy wybór każdego z rodziców polega na sprawdzeniu do którego przedziału wpadła losowo wybrana liczba.

Jak przydzielać przedziały chromosomom $x_1, x_2, , x_i$? Obliczmy $u = \min_{j=1,\dots,i} f(x_j)$.

Następnie przyporządkujmy każdemu x_j liczbę $w_j = f(x_j) + |u| + 1$. Dzięki temu zabiegowi jesteśmy pewni że $\forall_{j=1,\dots,i} w_j > 0$. Obliczamy $W = \sum_{j=1,\dots,i}^{j} w_j$. Obliczamy q_j, dla każdego $j \cup \{0\}$ według wzoru

$$q_j = \begin{cases} 0 & dla \quad j = 0 \\ q_{j-1}\frac{w_j}{W} & dla \quad 1 < j < i \\ 1 & dla \quad j = i \end{cases}$$

Chromosom $j - ty$ jest wylosowany gdy losowa liczba należy do przedziału $[q_{j-1}, q_j)$ (wyjątek stanowi $j = i$, wtedy przedział ma postać $[q_{i-1}, q_i]$).

Następną fazą algorytmu jest **krzyżowanie**. Czynność ta polega na stworzeniu, na podstawie pary chromosomów rodzicielskich, pary chromosomów potomnych. W zależności od rozpatrywanego problemu czynność ta może wyglądać bardzo różnie. Najczęściej, gdy chromosom jest liczbą binarną, krzyżowanie polega na przecięciu chromosomów rodzicielskich w ustalonym punkcie. Chromosomy potomne są tworzone poprzez sklejenie części początkowej chromosomu A z częścią końcową chromosomu B, oraz części początkowej chromosomu B z częścią końcową chromosomu A. Miejsce przecięcia może być ustalane na stałe w połowie długości chromosomu, lub być wybierane za każdym razem losowo. Jeżeli populacja ma j chromosomów a w fazie elitaryzmu zachowaliśmy e chromosomów to krzyżowania należy wykonać $\frac{j-e}{2}$ razy. Dzięki temu nie zmienimy znacząco liczebności populacji w trakcie działania algorytmu.

Po krzyżowaniu należy jeszcze dokonać **mutacji** chromosomów. Dla niewielkiej losowo wybranej liczby chromosomów zmieniamy losowo wybrane geny. Dzięki temu zabiegowi w populacji mają szansę pojawić się chromosomy, które nie miały by szansy pojawić się w wyniku selekcji, krzyżowania i elitaryzmu. Dzięki mutacji algorytm "nie grzęźnie".

Na koniec pętli należy obliczyć wartość funkcji przystosowania dla każdego nowo otrzymanego chromosomu.

Za warunek stopu, w pseudokodzie, przyjęte zostało wykonanie określonej liczby pętli, może też nim być również sytuacja w której przez kilka pokoleń nie pojawiło się lepsze rozwiązanie.

Jak może wyglądać algorytm genetyczny dotyczący szukania minimum kosztu wykonania planu lekcji realizującego przydział Π? Faza inicjacji i-elementowej populacji polega na wykonaniu i razy algorytmu *Generuj plan MC* 16. Następnie dla każdego otrzymanego planu należy obliczyć koszt jego wykonania. W fazie elitaryzmu należy przenieść do nowo tworzonej populacji od kilku do kilkunastu planów. Więcej uwagi należy poświęcić fazie krzyżowania. Można ją wykonać na co najmniej dwa sposoby:

- Mając dane plany A i B oraz macierze dyspozycji DN i DS, które można interpretować jako dwuwierszowe macierzą. (Jeden z wierszy odpowiadają za czas przeprowadzenia lekcji a drugi za przydzieloną salę.) Plany potomne A' i B' można utworzyć poprzez przepisanie pierwszego wiersza planu A oraz wiersza drugiego z B do A' oraz drugiego wiersza z A i pierwszego z B do planu B'.

Przykład 6.1. *Mając dane chromosomy rodzicielskie, czyli plany lekcji*

$$A = \begin{array}{r|ccccc|} \textbf{\textit{Czas}} & 1 & 2 & 2 & 1 & 2 \\ \textbf{\textit{Sala}} & 1 & 1 & 2 & 3 & 3 \end{array}$$

oraz

$$B = \begin{array}{r|ccccc|} \textbf{\textit{Czas}} & 1 & 2 & 2 & 1 & 1 \\ \textbf{\textit{Sala}} & 1 & 2 & 1 & 2 & 3 \end{array}$$

tworzymy chromosomy potomne w postaci:

$$A' = \begin{array}{r|ccccc|} \textbf{\textit{Czas z A}} & 1 & 2 & 2 & 1 & 2 \\ \textbf{\textit{Sala z B}} & 1 & 2 & 1 & 2 & 3 \end{array}$$

oraz

$$B' = \begin{array}{r|ccccc|} \textbf{\textit{Czas z B}} & 1 & 2 & 2 & 1 & 1 \\ \textbf{\textit{Sala z A}} & 1 & 1 & 2 & 3 & 3 \end{array}$$

Tak powstałe chromosomy potomne należy sprawdzić pod kątem konfliktu sal. Może się okazać, że w powstałym planie jest przydzielona różnym lekcjom, przeprowadzanym w tym samym czasie ta sama sala. Oczywiście taki plan należy odrzucić. Sprawdzenia można dokonać poprzez aktualizację macierzy DN i DS.

W przypadku zauważenia konfliktu pomiędzy dwoma lekcjami zapisanymi w kolumnach i oraz j można jedną lekcję zmodyfikować (zmienić jej czas lub przydzieloną salę) tak aby owy konflikt już nie zachodził. Można to wykonać przez losowe usunięcie lekcji i-tej lub $j-$tej oraz dodanie jej na nowo przy pomocy procedury *Generuj plan rekurencyjnie* 6 lub *Generuj plan rekurencyjnie MC* 15. Zadanie to wykonują procedury *Napraw* 19 oraz *Napraw MC* 20. Ten sposób rozwiązywania powstałych konfliktów można uznać za fazę **mutacji**. Nie jest ona przeprowadzana w zupełnie losowych przypadkach, z ustalonym prawdopodobieństwie zajścia (tak jaka ma to miejsce w typowych algorytmach genetycznych), lecz w miarę potrzeby.

- Mając dane plany A i B oraz wyjściowe tabele dyspozycyjności nauczycieli i sal, można utworzyć plan potomny w następujący sposób: dla każdej z lekcji występującej w przydziale losowo uzgadniamy czy najpierw próbujemy przypisać jej salę i czas z planu A czy też z planu B. Jeżeli natomiast, z powodów konfliktów, nie uda się do nowego planu skopiować ani lekcji z A ani z B, to wtedy dopisujemy daną lekcję do nowego planu na podstawie procedury *Generuj plan rekurencyjnie* 6 lub *Generuj plan rekurencyjnie MC* 15. Analogicznie do powyższego sposobu fazę dodawania, poprzez procedury *Generuj plan rekurencyjnie* i *Generuj plan rekurencyjnie MC*, uznać należy za fazę **mutacji**.

Pierwszy sposób przeprowadzenia krzyżowania jest podejściem wierszowym, natomiast sposób drugi działa kolumnowo. Niestety obydwa sposoby krzyżowania mogą doprowadzić do powstania konfliktów. Zaproponowany sposób rozwiązywania konfliktów może w niektórych przypadkach nie zadziałać. Dzieje się tak gdy dodanie konfliktowej lekcji nie jest możliwe, ponieważ lekcje o już ustalonym czasie i miejscu, powodują brak możliwości znalezienia czasu lub sali dla dodawanej lekcji. Krzyżowanie pierwszym sposobem można przedstawić w formie pseudokodu *Krzyżowanie I* 18 a sposób drugi w formie pseudokodu *Krzyżowanie II* 21.

Algorithm 18 Krzyżowanie I

Require: A, B, DN, DS.
Ensure: $W1$, $W2$
  $W1$, $W2 \leftarrow$ *pusty plan*
  for $i = 1 \to p$ **do**
    $(k, l) \leftarrow A[i]$
    $(m, n) \leftarrow B[i]$
    $W1[i] \leftarrow (k, n)$
    $W2[i] \leftarrow (n, l)$
    $W1 \leftarrow Napraw\ I(W1, DN, DS)$
    $W2 \leftarrow Napraw\ I(W2, DN, DS)$
    return $W1$, $W2$
  end for

W ostatnim kroku pętli należy obliczyć wartość funkcji kosztu dla nowo-powstałych planów oraz uszeregować je rosnąco według tej wartości.

Algorithm 19 Napraw

Require: W, Π, p, DN, DS.

Ensure: W,

```
Konflikty ← pusta lista
for i = 1 → p − 1 do
    (c_i, s_i) ← W[i]
    (n_i, p_i, k_i) ← Π[i]
    for j = i + 1 → p do
        (c_j, s_j) ← W[i]
        (n_j, p_j, k_j) ← Π[i]
        if c_i = c_j ∧ s_i = s_j then
            r ← rand(0, 1)
            if r = 1 then
                Konflikty ← Konflikty ∪ (n_i, p_i, k_i)
                W ← W\{(c_i, s_i)}
            else
                Konflikty ← Konflikty ∪ (n_j, p_j, k_j)
                W ← W\{(c_j, s_j)}
            end if
        end if
    end for
    if Warunek Konieczny Istnienia Planu = True then
        k ← dlugosc Konflikty
        p ← Generuj plan rekurencyjnie(Konflikty,k, p, DN, DS, DK)
        return p
    else
        Pisz "Nie można stworzyć planu"
    end if
end for
```

Algorithm 20 Napraw MC

Require: W, Π, p, DN, DS.

Ensure: W,

```
Konflikty ← pusta lista
for i = 1 → p − 1 do
    (c_i, s_i) ← p[i]
    (n_i, p_i, k_i) ← Π[i]
    for j = i + 1 → p do
        (c_j, s_j) ← p[i]
        (n_j, p_j, k_j) ← Π[i]
        if c_i = c_j ∧ s_i = s_j then
            r ← rand(0, 1)
            if r = 1 then
                Konflikty ← Konflikty ∪ (n_i, p_i, k_i)
                p ← p\{(c_i, s_i)}
            else
                Konflikty ← Konflikty ∪ (n_j, p_j, k_j)
                p ← p\{(c_j, s_j)}
            end if
        end if
    end for
    if Warunek Konieczny Istnienia Planu = True then
        k ← dlugosc Konflikty
        p ← Generuj plan rekurencyjnie MC(Konflikty,k, p, DN, DS, DK)
        return p
    else
        Pisz ”Nie można stworzyć planu”
    end if
end for
```

Algorithm 21 Krzyżowanie II

Require: Π, A, B,p, DN, DS.

Ensure: W

$W \leftarrow pusty\ plan$

$Konflikty \leftarrow pusta\ lista$

for $i = 1 \rightarrow p$ **do**

  $r \leftarrow rand(0, 1)$

  if $r = 0$ **then**

    $(c_1, s_1) \leftarrow A[i]$

    $(c_2, s_2) \leftarrow B[i]$

  else

    $(c_1, s_1) \leftarrow B[i]$

    $(c_2, s_2) \leftarrow A[i]$

  end if

  $(n, p, k) \leftarrow \Pi[i]$

  if $DN_{n,c_1} = 1 \wedge DS_{s,c_1} = 1$ **then**

    $W[i] \leftarrow (c_1, s_1)$

    $DN_{n,c_1} = 0$

    $DS_{s,c_1} = 0$

  else

    if $DN_{n,c_2} = 1 \wedge DS_{s,c_2} = 1$ **then**

      $W[i] \leftarrow (c_2, c_2)$

      $DN_{n,c_1} = 0$

      $DS_{s,c_1} = 0$

    else

      $Konflikty \leftarrow Konflikty \cup (n, p, k)$

    end if

  end if

end for

if Warunek Konieczny Istnienia Planu $= True$ **then**

  $k \leftarrow dlugosc\ Konflikty$

  $p \leftarrow$ Generuj plan rekurencyjnie MC($Konflikty$,k, p, DN, DS, DK)

  return p

else

  Pisz "Nie można stworzyć planu"

end if

Rozdział 7

Podsumowanie

W pierwszym rozdziale pracy zostało zaprezentowanych kilka modeli matematycznych planu lekcji. Ewoluowały one w kierunku modelu najoszczędniejszego w opisie i najwygodniejszego w implementacji. W dalszej części pracy stosowany jest tylko ostatni z nich. Następnie omówione zostały kwestie poprawności. Zwieńczeniem rozdział drugiego jest matematyczna definicja planu poprawnego.

Tematyką następnego rozdziału jest kwestia istnienia planu lekcji. Efektem prac przeprowadzonych w tym rozdziale jest kilku warunków koniecznych istnienia planu, stanowiących bardzo ważny element dwóch wyprowadzonych algorytmów generujących plan. Stworzone algorytmy nie zapewniają sukcesu w ułożeniu poprawnego planu. Pesymistyczny czas ich działania wynosi $O(n!)$ gdzie n jest liczbą lekcji w tworzonym planie, jednak czas optymistyczny oraz średni jest rzędu $O(n)$. Algorytm *Generuj plan* jest tak skonstruowany aby dawać w wyniku "pierwszy z brzegu" istniejący plan. Algorytm *Generuj plan według dyspozycji* wymaga natomiast większej pracy ze strony programisty oraz sprzętu. Algorytm ten stara się pracować z lekcjami nauczyciela najbardziej zapracowanego czyli z lekcjami które jest najtrudniej wstawić w plan. Dodaje on kłopotliwe lekcje w tworzony plan, w pierwszej kolejności, dzięki czemu algorytm nie jest zmuszany wykonywać wielu nawrotów. Efektem takiego podejścia jest bardzo dobry czas pracy algorytmu. Plany mające do 300 lekcji, są przez obydwa algorytmy, układane w czasie poniżej 1 sekundy.

W rozdziale piątym wyprowadzona została funkcja celu. Jej konstrukcja uwzględnia liczbę okienek nauczyciela, liczbę potrzebnych sal oraz liczbę zmian sali. Funkcja ta jest bardzo podatna na korekty planu: zmiana czasu przeprowadzenia lekcji lub jednej sali może silnie oddziaływać na jej wartość, co powoduje, że nie jest to funkcja łatwa do optymalizacji. Własności funkcji celu skłaniają do stosowania przede wszystkim losowych algorytmów optymalizacyjnych.

Tematyką kolejnej części pracy są dwa losowe algorytmy optymalizacyjne. Jednym z nich jest algorytm typu *Monte Carlo* Opiera się on na procedurze dodającej lekcję o losowym czasie i losowej sali do tworzonego planu. Wspomniana procedura działa w czasie liniowym.

Drugim algorytmem losowym jest *Algorytm genetyczny*. W jego implementacji uwzględniono dwie metody krzyżowania planów. Pierwsza z nich polega na przepisaniu do tworzonego planu czasów prowadzenia lekcji od pierwszego rodzica oraz sal od drugiego rodzica. Drugi sposób krzyżowania polega, w miarę możliwości, na kopiowaniu po jednej lekcji od losowo wybranego rodzica. W samym algorytmie, jak i w krzyżowaniu, często stosowane są algorytmy sortowania, w wyniku czego, krzyżowanie wykonuje się w czasie $O(n \log n)$. Metoda ta wymaga dużo większego wysiłku w implementacji.

W tym samym rozdziale udało się stworzyć algorytm zachłanny generujący plan bliski planowi optymalnemu. Wymaga on największego nakładu pracy programisty w implementacji: często dochodzi w nim do sortowań oraz do wielu wyliczeń różnych współczynników. Algorytm ten jest dobrym rozwiązaniem w przypadku dużych zestawów danych. Pozwala on, szybciej niż wspomniane algorytmy losowe, na konstrukcję rozwiązania bliskiego optymalnemu.

Każdy z omawianych algorytmów został zaimplementowany oraz przetestowany. Dla przydziałów poniżej 300 lekcji w sytuacji gdy każdy nauczyciel mógł pracować w każdym czasie, najmniejsza wartość funkcji celu osiągana była dla planu będącego wynikiem pracy procedury *Generuj plan*. Niewiele gorsze efekty dawały procedury *Generuj plan według dyspozycji* oraz *Algorytm zachłanny*. Wartość funkcji celu, obliczana na podstawie wygenerowanego planu, była większa o około 4%. Procedury te działały w podobnym czasie (około jednej sekundy). Natomiast algorytmy losowe dawały dużo gorszy wynik w dużo gorszym czasie. Efekt działania procedury *Generuj plan* jest najlepszy ponieważ w sytuacji gdy każdy może pracować w każdym czasie, dodawanie lekcji w pierwszym wolnym czasie, w pierwszej wolnej sali generuje najmniej kosztów.

W sytuacji gdy nauczyciele nie byli dyspozycyjni w około 80% swojego czasu, prawie wszystkie algorytmy osiągały podobny wynik. Wyjątkiem był *Algorytm zachłanny* który dla dużych planów częściej kończył się błędem. Natomiast czasy ich działania nie różnił się od przypadku poprzedniego.

W sytuacji gdy nauczyciele nie byli dyspozycyjni w około 60% swojego czasu, lepsze wyniki osiągały algorytmy losowe, jednak *Algorytm Monte Carlo* wielokrotnie dłuższym czasie.

W optymalizacji planu lekcji, wybór najlepszej metody zależy od dyspozycyjności nauczycieli. Czym ona jest mniejsza tym lepiej wypadają algorytmy genetyczne.

Dodatek A

Wyniki testów

Wyniki testów przeprowadzonych w sytuacji gdy każdy nauczyciel i każda sala są dostępne.

Tabela A.1: *Generuj plan*

Liczba lekcji	100	200	300
Czas działania	$< 1s$	1s	1s
osiągnięte minimum	23.1	28	41.8

Tabela A.2: *Generuj plan według dyspozycji*

Liczba lekcji	100	200	300
Czas działania	$< 1s$	1s	1s
osiągnięte minimum	25.3	29.7	42.9

Tabela A.3: *Algorytm zachłanny*

Liczba lekcji	100	200	300
Czas działania	$< 1s$	1s	1s
osiągnięte minimum	24.2	29.7	Porażka. Wygenerowano 98% planu

Tabela A.4: *Algorytm Monte Carlo* dla 10000 prób.

Liczba lekcji	100	200	300
Czas działania	6s	21s	52s
osiągnięte minimum	22	30.8	47.3

Tabela A.5: *Algorytm genetyczny* z krzyżowaniem I dla 10 pokoleń z 1000 elementów w populacji.

Liczba lekcji	100	200	300
Czas działania	2s	4s	14s
osiągnięte minimum	23.1	30.8	47.3

Tabela A.6: *Algorytm genetyczny* z krzyżowaniem II dla 10 pokoleń z 1000 elementów w populacji.

Liczba lekcji	100	200	300
Czas działania	4s	7	14s
osiągnięte minimum	23.1	30.8	47.3

Tabela A.7: Wyniki testów przeprowadzonych w sytuacji gdy nauczyciele są dyspozycyjni w 80%.

Liczba lekcji	100	200	300
Algorytm zachłanny	24.2	31.9	Porażka. Wygenerowano 99% planu
Algorytm Monte Carlo	23.1	31.9	47.3 (76 s)
Algorytm genetyczny z krzyżowaniem I	23.1	30.8	47.3
Algorytm genetyczny z krzyżowaniem II	22	30.8	47.3

Tabela A.8: Wyniki testów przeprowadzonych w sytuacji gdy nauczyciele są dyspozycyjni w 60%.

Liczba lekcji	100	200	300
Algorytm zachłanny	24.2	31.9	47.3
Algorytm Monte Carlo	23.1	30.8	47.3 (6 min 12s)
Algorytm genetyczny z krzyżowaniem I	23.1	30.8	47.3 (47s)
Algorytm genetyczny z krzyżowaniem II	23.1	31.9	47.3 (1 min 31 s)

Dodatek B

Stosowane oznaczenia

Nazwa zbioru	Oznaczenie zbioru	Oznaczenie mocy zbioru	Oznaczenie elementu zbioru
Zbiór klas	K	k	q
Zbiór nauczycieli	N	n	t
Zbiór przedmiotów	P	p	d
Zbiór sal	S	s	
Zbiór czasów rozpoczęcia lekcji	C	c	
Zbiór nauczycieli rozszerzony o pusty element	$N' = N \cup \{\epsilon\}$		
Zbiór przedmiotów rozszerzony o pusty element	$P' = P \cup \{\epsilon\}$		
Zbiór sal rozszerzony o pusty element	$N' = S \cup \{0\}$		

PL_1 - plan lekcji typu I

$PL_2(K, C)$ - plan lekcji typu II

DN_{nc} - zero-jedynkowa macierz dyspozycji nauczycieli o wymiarach n na c

DS_{ns} - zero-jedynkowa macierz dostępności sal o wymiarach s na c

Bibliografia

[1] OPTYMALIZACJA Wybrane metody z przykładami zastosowań *Jan Kusiak, Anna Danielewska-Tułecka, Piotr Oprocha.* PWN Warszawa 2009.

[2] Wprowadzenie do algorytmów Wydanie czwarte *Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest*

[3] School timetable construction Algorithms and complexity *Robertus Johannes Willemen* 2002