Politechnika Krakowska
im. Tadeusza Kościuszki

Wydział
Inżynierii Lądowej

ROZPRAWA DOKTORSKA

mgr inż. Bartłomiej Sroka

METODA PRIORYTETOWEGO HARMONOGRAMOWANIA WIELOOBIEKTOWYCH PRZEDSIĘWZIĘĆ BUDOWLANYCH

Promotor:
prof. dr hab. inż. Elżbieta Radziszewska-Zielina

Kraków 2023

Spis treści

Ważniejsze oznaczenia stosowane w rozprawie 5

1. Wstęp 10

1.1. Wprowadzenie w problematykę rozprawy 10

1.2. Zakres i schemat pracy 12

1.3. Stosowane metody i narzędzia badawcze 14

2. Uzasadnienie tematu rozprawy 17

2.1. Wstępne badania kwestionariuszowe 17

2.2. Analiza rynku wieloobiektowych przedsięwzięć budowlanych 20

2.3. Przegląd literatury polskiej i zagranicznej 25

2.3.1. Nomenklatura występująca w harmonogramowaniu przedsięwzięć wieloobiektowych 26

2.3.2. Główne kierunki rozwoju metod harmonogramowania wieloobiektowych przedsięwzięć budowlanych 28

2.3.3. Metody uwzględniające analizę czasową 30

2.3.4. Metody uwzględniające analizę kosztową 35

2.3.5. Prace doktorskie i monografie 42

2.3.6. Analiza liczby publikacji w bazie Scopus 44

2.4. Podsumowanie uzasadnienia wyboru tematu 45

2.5. Teza badawcza 47

2.6. Cele rozprawy doktorskiej 47

3. Metoda priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych 48

3.1. Specyfika harmonogramowania wieloobiektowych przedsięwzięć budowlanych 48

3.2. Koncepcja harmonogramowania priorytetowego 52

3.3. Opis opracowanej metody 62

3.3.1. Moduł decydenta 63

3.3.2. Czasowo-kosztowy model priorytetowego harmonogramowania 67

3.3.3. Zlinearyzowany model czasowo-kosztowy 72

3.3.4. Metoda optymalizacji dyskretnej uszeregowania obiektów 75

3.3.5. Moduł użytkowy 78

3.4. Opracowane zestawy wag realizujące różne założenia decyzyjne 79
3.4.1. Zestawy wag uwzględniające sprzężenia elastyczne klasyczne 80
3.4.2. Zestawy wag uwzględniające sprzężenia elastyczne dodatkowe 82
3.4.3. Zestawy wag uwzględniające sprzężenia elastyczne niestandardowe 85
4. Przykłady obliczeniowe 87
4.1. Przykład zastosowania zlinearyzowanego modelu czasowo-kosztowego 87
4.2. Przykład zastosowania modelu harmonogramowania priorytetowego 89
4.3. Wybór optymalnych parametrów oraz przykład zastosowania modułu MCTS 97
4.4. Przykład zastosowania modułu użytkowego 104
4.5. Przykład kompleksowy 105
4.5.1. Przykład 1 105
4.5.2. Przykład 2 111
4.5.3. Przykład 3 114
4.6. Weryfikacja i walidacja opracowanego modelu 116
4.6.1. Walidacja konceptualna modelu 116
4.6.2. Skomputeryzowana weryfikacja modelu 116
4.6.3. Walidacja operacyjna modelu 116
4.6.4. Walidacja danych 120
5. Podsumowanie 121
5.1. Wnioski końcowe 121
5.2. Kierunki dalszych badań 122
6. Bibliografia 124
7. Załączniki 133
8. Spis tabel, rysunków i załączników 186
8.1. Spis rysunków 186
8.2. Spis tabel 188
8.3. Spis załączników 190
9. Streszczenie 191
10. Abstract 192

Ważniejsze oznaczenia stosowane w rozprawie

$a_{i,j,k}$ – współczynnik kierunkowy zlinearyzowanej funkcji kosztów od czasu w przedziale określonym przez $t_{i,j,s}$

$\hat{B}$- zbiór par indeksów pomocny przy określaniu sprzężeń czasowych pomiędzy procesami wykonywanymi przez różne brygady

$\bar{\bar{B}}$ – zbiór numerów brygad, dla których ma być zapewniona ciągłość realizacji procesu

Bss – suma liczby symulacji dla najlepszego węzła poniżej węzła stanowiącego korzeń (root) drzewa w metodzie MCTS

BV – najlepsza odnaleziona wartość funkcji celu w metodzie MCTS

BV_i – najlepsza odnaleziona wartość funkcji celu dla węzła i w metodzie MCTS

C – parametr odpowiedzialny za równowagę eksploatacji i eksploracji w metodzie MCTS

C_p^A – dodatkowe elastyczne sprzężenie czasowe

$C_{i,j}^B$ – elastyczne sprzężenia czasowe pomiędzy procesami wykonywanymi przez kolejne brygady na jednym obiekcie

$C_{i,j}^O$ – elastyczne sprzężenia czasowe pomiędzy procesami na kolejnych obiektach wykonywanych przez tę samą brygadę

CK_p^{AL} – zmienna pomocnicza w modelu harmonogramowania priorytetowego – pozwala wyznaczyć niedotrzymanie wartości dolnego dodatkowego elastycznego sprzężenia czasowego

CK_p^{AU} – zmienna pomocnicza w modelu harmonogramowania priorytetowego – pozwala wyznaczyć niedotrzymanie wartości górnego dodatkowego elastycznego sprzężenia czasowego

$CK_{i,j}^{BL}$ – zmienna pomocnicza w modelu harmonogramowania priorytetowego – pozwala wyznaczyć niedotrzymanie wartości dolnej elastycznego sprzężenia czasowego pomiędzy procesami wykonywanymi przez kolejne brygady

$CK_{i,j}^{BU}$ – zmienna pomocnicza w modelu harmonogramowania priorytetowego – pozwala wyznaczyć niedotrzymanie wartości górnej elastycznego sprzężenia czasowego pomiędzy procesami wykonywanymi przez kolejne brygady

$CK_{i,j}^{OL}$ – zmienna pomocnicza w modelu harmonogramowania priorytetowego – pozwala wyznaczyć niedotrzymanie wartości dolnej elastycznego sprzężenia czasowego pomiędzy procesami wykonywanymi na kolejnych obiektach

$CK_{i,j}^{OU}$ – zmienna pomocnicza w modelu harmonogramowania priorytetowego – pozwala wyznaczyć niedotrzymanie wartości górnej elastycznego sprzężenia czasowego pomiędzy procesami wykonywanymi na kolejnych obiektach

CT_p^{AL} – wartość dolnego dodatkowego elastycznego sprzężenia czasowego pomiędzy odpowiednimi charakterystykami procesów S_p oraz P_p

CT_p^{AU} - wartość górnego dodatkowego elastycznego sprzężenia czasowego pomiędzy odpowiednimi charakterystykami procesów S_p oraz P_p

$CT_{i,j}^{BL}$ – wartość dolna elastycznego sprzężenia czasowego pomiędzy procesami realizowanymi przez kolejne brygady na jednym obiekcie

$CT_{i,j}^{BU}$ – wartość górna elastycznego sprzężenia czasowego pomiędzy procesami realizowanymi przez kolejne brygady na jednym obiekcie

$CT_{i,j}^{OL}$ – wartość dolna elastycznego sprzężenia czasowego pomiędzy procesami na kolejnych obiektach realizowanych przez tę samą brygadę

$CT_{i,j}^{OU}$ – wartość górna elastycznego sprzężenia czasowego pomiędzy procesami na kolejnych obiektach realizowanych przez tę samą brygadę

CW_p^{AL} – wartość jednostkowa kary za niedotrzymanie sprzężenia CT_p^{AL}

CW_p^{AU} - wartość jednostkowa kary za niedotrzymanie sprzężenia CT_p^{AU}

$CW_{i,j}^{BL}$ – jednostkowa kara za niedotrzymanie wartości dolnej sprzężenia $CT_{i,j}^{BL}$

$CW_{i,j}^{BU}$ – wartość jednostkowa kary za niedotrzymanie sprzężenia $CT_{i,j}^{BU}$

$CW_{i,j}^{OL}$ – jednostkowa kara za niedotrzymanie wartości dolnej sprzężenia $CT_{i,j}^{OL}$

$CW_{i,j}^{OU}$ – jednostkowa kara za niedotrzymanie wartości górnej sprzężenia $CT_{i,j}^{OU}$

$d_{k,i}$ – zmienne linearyzujące zależność kosztu od czasu dla procesu *k* w zlinearyzowanym modelu czasowo-kosztowym

$d_{i,j,k}$ - zmienna linearyzująca zależność kosztu od czasu realizacji procesu

$K(\pi)$ – koszty całkowite przedsięwzięcia wieloobiektowego przy uszeregowaniu π

$\overline{K(\pi)}$ – wartość funkcji celu dla przedsięwzięcia wieloobiektowego przy uszeregowaniu π

K_{bez} – całkowity koszt bezpośredni

K_c – koszty nieciągłości pracy brygad roboczych

kc_j – jednostkowe koszty za niedotrzymanie ciągłości pracy brygady *j*

K_{CA} – koszt niedotrzymania dodatkowych elastycznych sprzężeń czasowych

K_{CB} – koszt niedotrzymania elastycznych sprzężeń czasowych pomiędzy procesami wykonywanymi przez kolejne brygady na jednym obiekcie

K_{CO} – koszt niedotrzymania elastycznych sprzężeń czasowych pomiędzy procesami na kolejnych obiektach wykonywanych przez tę samą brygadę

$kgr_{i,j}$ – koszt graniczny procesu wykonywanej na obiekcie *i* przez brygadę *j*

K_n – premie za wcześniejsze zrealizowanie obiektów

kn_i – jednostkowa premia za wcześniejsze wykonanie obiektu *i*

K_p – koszt niedotrzymania terminów dyrektywnych zrealizowania obiektów

kp_i – jednostkowe koszty za niedotrzymanie terminu dyrektywnego na obiekcie *i*

$K_{poś}$ – całkowity koszt pośredni

$k_{poś}$ – jednostkowe koszty pośrednie przedsięwzięcia wieloobiektowego

KT – składnik w funkcji celu, która pozwala na wyznaczenie terminów najwcześniejszych oraz najpóźniejszych procesów

$\widetilde{L_1}, \widetilde{L_2}, \dots, \widetilde{L_k}$ – dostatecznie duże liczby, takie, że: $\widetilde{L_1} \ll \widetilde{L_2} \ll \cdots \ll \widetilde{L_k}$

m – liczba brygad przeznaczonych do wykonania przedsięwzięcia wieloobiektowego

n – liczba obiektów przedsięwzięcia wieloobiektowego

$\hat{n}$ – liczba obiektów, dla których można przeprowadzić przegląd zupełny wszystkich uszeregowań

n_i - wartość wcześniejszego wykonania obiektu *i* w stosunku do terminu dyrektywnego

$NPR_{i,j}$ – najpóźniejszy termin rozpoczęcia procesu (*i,j*)

$NPZ_{i,j}$ – najpóźniejszy termin zakończenia procesu (*i,j*)

$NPZ_{i,m}$ – termin (najpóźniejszy) zakończenia realizacji procesu przez ostatnią brygadę na obiekcie *i*

$NPZ_{n,j}$ – termin (najpóźniejszy) zakończenia realizacji procesu na ostatnim obiekcie przez brygadę *j*

$NPZ_{n,m}$ –termin (najpóźniejszy) zakończenia przedsięwzięcia wieloobiektowego

$NWR_{1,j}$ – termin (najwcześniejszy) rozpoczęcia realizacji procesu na pierwszym obiekcie przez brygadę *j*

$NWR_{i,1}$ – termin (najwcześniejszy) rozpoczęcia realizacji procesu przez pierwszą brygadę na obiekcie *i*

$NWR_{i,j}$ – najwcześniejszy termin rozpoczęcia procesu (*i,j*)

$NWZ_{i,j}$ – najwcześniejszy termin zakończenia procesu (*i,j*)

$NWZ_{n,m}$ –termin (najwcześniejszy) zakończenia przedsięwzięcia wieloobiektowego

$\hat{O}$ – zbiór par indeksów pomocny przy określaniu sprzężeń czasowych pomiędzy procesami na różnych obiektach

$\bar{\bar{O}}$ – zbiór numerów obiektów, na których ma być zapewniona ciągłość realizacji procesu

$\overline{P_1}, \overline{P_2} \ldots \overline{P_k}$ – zbiór priorytetów decydenta w czasowo-kosztowym modelu priorytetowego harmonogramowania

p_i - wartość opóźnienia w stosunku do terminu dyrektywnego na obiekcie *i*

P_p – odpowiednia charakterystyka następnika dodatkowego sprzężenia czasowego proces. $P_p \in \{NWR_{k,l}; NWZ_{k,l}; NPR_{k,l}; NPZ_{k,l}; ZC_{k,l}\}$

Pss_i – suma liczby symulacji wszystkich następników dla bezpośredniego poprzednika węzła i w metodzie MCTS

Rss – suma liczby symulacji dla węzła stanowiącego korzeń (root) drzewa w metodzie MCTS

s_i – liczba wszystkich symulacji dla węzła i w metodzie MCTS

S_i – sposób wyboru węzła w metodzie MCTS

S_p – odpowiednia charakterystyka następnika dodatkowego sprzężenia czasowego procesu. $S_p \in \{NWR_{i,j}; NWZ_{i,j}; NPR_{i,j}; NPZ_{i,j}; ZC_{i,j}\}$

$\sum w_i$ – suma wszystkich uzyskanych wartości funkcji celu dla węzła i w metodzie MCTS

$\sum ss_i$ – liczba symulacji zakończonych sukcesem dla węzła i w metodzie MCTS

Td_i – termin dyrektywny zakończenia prac na obiekcie *i*

T_j^{FB} – najpóźniejszy termin dostępności brygady *j*

T_i^{FO} – najpóźniejszy termin dostępności obiektu *i*

t_k – czas trwania procesu *k* w zlinearyzowanym modelu czasowo-kosztowym

T_j^{SB} – najwcześniejszy termin dostępności brygady *j*

T_i^{SO} – najwcześniejszy termin dostępności obiektu *i*

$tgr_{i,j}$ – czas graniczny procesu wykonywanego na obiekcie *i* przez brygadę *j*

$t_{i,j,k}$ – długość przedziału, na jakim obowiązuje zlinearyzowana funkcja kosztów od czasu

$t_{i,j}$ – czas trwania realizacji procesu (*i,j*)

T_{MCTS} – czas wykonania programu w metodzie MCTS

TZ – termin zakończenia całego przedsięwzięcia w zlinearyzowanym modelu czasowo-kosztowym

TZ_k – termin zakończenia procesu *k* w zlinearyzowanym modelu czasowo-kosztowym

V_i – sposób obliczenia wartości w węźle w metodzie MCTS

$\widehat{W}$ - zbiór par indeksów wszystkich procesów Modelu Sieciowego Przedsięwzięcia

$ZC_{i,j}$ – całkowity zapas czasu procesu (*i,j*)

Przyjęta konwencja:

$a \ll b$ – oznacza, że liczba a jest dużo mniejsza niż b

$a_1, a_2, \dots, a_k \geq 0$ – jest równoważne z: $a_1 \geq 0 \wedge a_2 \geq 0 \wedge \dots \wedge a_k \geq 0$

$a_1, a_2, \dots, a_k \leq 0$ – jest równoważne z: $a_1 \leq 0 \wedge a_2 \leq 0 \wedge \dots \wedge a_k \leq 0$

$a_1, a_2, \dots, a_k = 0$ – jest równoważne z: $a_1 = 0 \wedge a_2 = 0 \wedge \dots \wedge a_k = 0$

$a \prec b$ – a poprzedza b

1. Wstęp

1.1. Wprowadzenie w problematykę rozprawy

Poprzez wieloobiektowe przedsięwzięcie budowlane należy rozumieć takie przedsięwzięcie budowlane, podczas realizacji którego do wykonania jest więcej niż jeden obiekt budowlany. Każdy obiekt realizowany podczas przedsięwzięcia może być podzielony na działki robocze. W niniejszej rozprawie wieloobiektowe przedsięwzięcia budowlane będą również nazywane przedsięwzięciami wieloobiektowymi.

W ostatnim czasie można zauważyć rosnącą liczbę przedsięwzięć wieloobiektowych. Są to głównie przedsięwzięcia polegające na budowie osiedli mieszkaniowych jedno- i wielorodzinnych, kompleksy bloków, kompleksy biurowców lub obiekty użyteczności publicznej. Przykładami takich realizacji mogą być: Osiedle Avia, Osiedle Bagry Park, Osiedle Sosnowiecka, szpital Uniwersytecki w Prokocimiu, High Five przy ul. Pawiej lub O3 Business Campus przy ul. Opolskiej, położone w Krakowie. Przedsięwzięcia tego typu realizowane są na całym świecie.

Z uwagi na fakt, iż przedsięwzięcia wieloobiektowe charakteryzują się dużą pracochłonnością, zazwyczaj są one realizowane w systemie generalnego wykonawstwa, co przysparza wielu problemów natury organizacyjnej. Zatrudnianie podwykonawców powoduje, że na placu budowy pracuje wiele brygad, których działanie koordynuje generalny wykonawca. Umowy zawarte pomiędzy generalnym wykonawcą, a podwykonawcami określają terminy, w jakich dany podwykonawca powinien wykonywać prace. Przy przedsięwzięciach wieloobiektowych, realizowanych przez dużą liczbę podwykonawców, unikanie konfliktów jest kwestią problematyczną. Podczas realizacji przedsięwzięcia wieloobiektowego może nastąpić sytuacja, w której uniknięcie wszystkich konfliktów nie jest możliwe. Należy wtedy minimalizować liczbę tych konfliktów. Zasadnym jest modelowanie ograniczeń, nie w sposób sztywny, ale elastyczny, czyli taki, który pozwoli dotrzymywać ograniczenia w możliwie największym stopniu. Ustalenie przez kierownika budowy terminów rozpoczęcia oraz zakończenia prac przez poszczególnych podwykonawców jest kluczowe w celu zapewnienia prawidłowej, bezkonfliktowej realizacji przedsięwzięcia wieloobiektowego.

Ponadto, przedsięwzięcia wieloobiektowe zazwyczaj są przedsięwzięciami, których koszty realizacji są wysokie. Przykładowo, koszt budowy nowego szpitala uniwersyteckiego w Prokocimiu wyniósł (wraz z wykończeniem i wyposażeniem) około 1,2 miliarda złotych. Generalny wykonawca musi więc uważać, aby nie ponosić dodatkowych kosztów związanych z podpisanymi umowami z inwestorem oraz podwykonawcami. Koszty niedotrzymania terminów dyrektywnych realizacji poszczególnych obiektów, koszty niezapewnienia frontu roboczego dla

podwykonawców, koszty związane z przyspieszeniem prac mogą znacząco wpłynąć na koszty poniesione przez generalnego wykonawcę, zmniejszając jego zyski. W skrajnych przypadkach niewłaściwe zarządzanie budową może przynieść straty dla generalnego wykonawcy.

Podczas projektowania realizacji przedsięwzięć wieloobiektowych należy również wziąć pod uwagę kolejność realizacji poszczególnych obiektów, ponieważ ma to wpływ na czas i koszt realizacji przedsięwzięcia. Kolejność realizacji obiektów zazwyczaj jest narzucona przez inwestora. Wynika to głównie z przyczyn marketingowych. Obiekty zrealizowane od strony ruchliwej ulicy mogą zachęcić do kupna potencjalnych nabywców. Niekiedy kolejność wykonywanych obiektów wynika z zastosowanej technologii oraz narzuconych przez projektanta rozwiązań. W każdej sytuacji należy jednak analizować efekty zmian w planowanej kolejności wykonania obiektów. Może to przynieść pozytywne skutki zarówno dla generalnego wykonawcy (np. dotrzymanie terminów dyrektywnych, mniejsze koszty wykonania prac), jak również dla inwestora (wcześniejsze oddanie budynku do użytkowania spowoduje, że lokale będą wcześniej wynajmowane lub sprzedane, co przyniesie dodatkowe zyski).

Mając na uwadze wspomniane powyżej problemy, niezbędnym wydaje się opracowanie metody pozwalającej na zwiększenie efektywności harmonogramowania realizacji przedsięwzięć wieloobiektowych. Uwzględnienie ograniczeń technologiczno-organizacyjnych, ponoszonych kosztów oraz kolejności wykonania obiektów umożliwi racjonalne harmonogramowanie realizacji przedsięwzięcia wieloobiektowego, zapewniając wymierne korzyści inwestorowi, generalnemu wykonawcy oraz podwykonawcom.

1.2. Zakres i schemat pracy

Na rysunku 1 przedstawiono graficznie schemat rozprawy zawierający najważniejsze jej elementy. Dodatkowo na schemacie znajdują się numery rozdziałów, w których można znaleźć informacje na dany temat. Strzałki oznaczają powiązania logiczne pomiędzy elementami rozprawy.

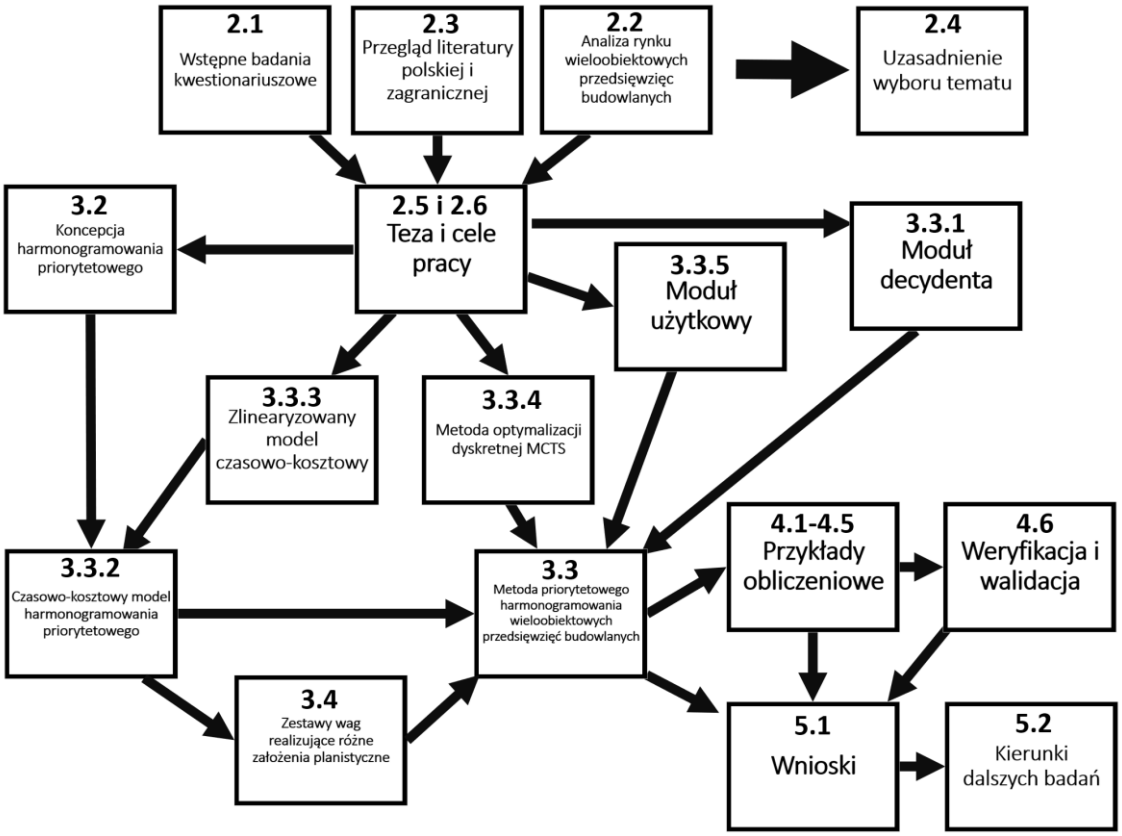
W rozdziale 1 wprowadzono w problematykę rozprawy, przedstawiono zakres i schemat pracy oraz opisano stosowane metody i narzędzia badawcze.

W rozdziale 2 uzasadniono temat rozprawy. Rozdział zawiera opis przeprowadzonych wstępnych badań kwestionariuszowych, w których respondentami byli specjaliści w zakresie kierowania i harmonogramowania przedsięwzięć wieloobiektowych oraz osoby decyzyjne w przedsiębiorstwach budowlanych. Dodatkowo przedstawiono sytuację na rynku wieloobiektowych przedsięwzięć budowlanych na przykładzie wyników finansowych trzech przedsiębiorstw deweloperskich oraz ogólnej sytuacji na rynku mieszkaniowym na podstawie danych GUS. W rozdziale zaprezentowano również przegląd literatury dotyczącej projektowania realizacji i harmonogramowania przedsięwzięć o charakterze powtarzalnym. Wyciągnięto wnioski z przeprowadzonych analiz oraz przedstawiono tezę i cele rozprawy.

W rozdziale 3 przedstawiono specyfikę harmonogramowania wieloobiektowego wraz z przyjętymi założeniami do opracowanej metody. Przedstawiono koncepcję harmonogramowania priorytetowego oraz metodę priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych. Omówiono każdy opracowany element metody. Przedstawiono opracowany zlinearyzowany model czasowo-kosztowy oraz opracowany czasowo-kosztowy model harmonogramowania priorytetowego. Podano również założenia i sposób działania opracowanego modułu optymalizacji dyskretnej uszeregowania obiektów, modułu decydenta oraz modułu użytkowego. Zaprezentowano opracowane zestawy wag, które pomagają decydentowi w modelowaniu swoich preferencji w kwestii ograniczeń technologiczno-organizacyjnych.

W rozdziale 4 pokazane zostało zastosowanie zlinearyzowanego modelu czasowo-kosztowego, czasowo-kosztowego modelu priorytetowego harmonogramowania, wybór optymalnych parametrów wraz z porównaniem z innymi metodami modułu optymalizacji dyskretnej uszeregowania obiektów, przykład zastosowania modułu użytkowego oraz trzy kompleksowe przykłady obliczone z zastosowaniem opracowanej metody harmonogramowania priorytetowego wieloobiektowych przedsięwzięć budowlanych. Przeprowadzono również weryfikację i walidację opracowanej metody.

W rozdziale 5 Zaprezentowano wnioski końcowe, kierunki dalszych badań, wartość naukową oraz wartość praktyczną rozprawy.



Rysunek 1: Schemat rozprawy

1.3. Stosowane metody i narzędzia badawcze

Do realizacji celów rozprawy doktorskiej posłużą następujące metody:

- **Wywiad swobodny**

  W celu uzasadnienia wyboru tematu oraz określenia problemów podczas planowania i realizacji przedsięwzięć wieloobiektowych dokonano wywiadów swobodnych z inżynierami, kierownikami robót oraz z prezesami i osobami decyzyjnymi w przedsiębiorstwach zajmujących się realizacją przedsięwzięć wieloobiektowych. W wyniku analizy przeprowadzonych wywiadów sformułowano wnioski w związku z problemami i brakami w planowaniu przedsięwzięć wieloobiektowych.

- **Przegląd literatury**

  Dokonano przeglądu literatury naukowej w ramach planowania i realizacji przedsięwzięć budowlanych, ze szczególnym uwzględnieniem planowania przedsięwzięć wieloobiektowych. Wyróżniono główne kierunki rozwoju metod harmonogramowania. Dokonano analizy artykułów naukowych polskojęzycznych oraz obcojęzycznych oraz dostępnych dysertacji i monografii z zakresu planowania przedsięwzięć wieloobiektowych. W ramach przeglądu literatury dokonano też analizy liczby publikacji w obszarze przedsięwzięć wieloobiektowych na przykładzie bazy Scopus.

- **Analiza danych**

  Dokonano analizy danych uzyskanych z trzech przedsiębiorstw zajmujących się realizacją przedsięwzięć wieloobiektowych. Przeanalizowano sytuację finansową oraz liczbę sprzedanych lokali przez te przedsiębiorstwa. Dodatkowo przeanalizowano dane uzyskane z Głównego Urzędu Statystycznego na temat liczby oddawanych lokali użytkowych, co daje obraz sytuacji na rynku przedsięwzięć wieloobiektowych.

- **Metoda Sprzężeń Czasowych**

  W oparciu o Metodę Sprzężeń Czasowych została opracowana koncepcja harmonogramowania priorytetowego. Został również opracowany czasowo-kosztowy model priorytetowego harmonogramowania, który pozwala wyznaczyć optymalne pod względem czasowo-kosztowym terminy rozpoczęcia i zakończenia procesów realizowanych podczas wieloobiektowego przedsięwzięcia budowlanego, z uwzględnieniem w sposób elastyczny ograniczeń organizacyjno-technologicznych.

- **Programowanie liniowe oraz algorytm Simplex**

  Opracowany czasowo-kosztowy model priorytetowego harmonogramowania jest optymalizacyjnym modelem liniowym, należy zatem do klasy problemów

programowania liniowego. Przedstawienie ograniczeń i funkcji celu w postaci liniowej jest uproszczeniem, natomiast ze względu na wykładniczą złożoność obliczeniową problemu wyboru kolejności obiektów jest podejściem właściwym. Algorytm Simplex posłużył znalezieniu optymalnego rozwiązania czasowo-kosztowego modelu priorytetowego harmonogramowania. Algorytm Simplex jest uważany za szybki, co przy wykładniczej złożoności obliczeniowej problemu wyboru optymalnej kolejności obiektów ma szczególne znaczenie.

- **Programowanie sieciowe**

  Przedsięwzięcie wieloobiektowe zostało przedstawione w postaci Modelu Sieciowego Przedsięwzięcia (MSP). Programowanie sieciowe pozwoliło na zamodelowanie zagadnienia wyznaczania terminów rozpoczęcia i zakończenia poszczególnych procesów oraz terminu realizacji MSP.

- **Monte Carlo Tree Search**

  Metoda optymalizacji dyskretnej Monte Carlo Tree Search (MCTS), której opracowana modyfikacja została wykorzystana do wyznaczania optymalnego uszeregowania obiektów w metodzie priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych.

- **Weryfikacja i walidacja**

  Dokonano wieloetapowej weryfikacji opracowanej metody w postaci weryfikacji konceptualnej modelu, skomputeryzowanej weryfikacji modelu, walidacji operacyjnej (w tym analizy wrażliwości) oraz walidacji danych.

Do realizacji celów rozprawy doktorskiej posłużył język programowania **Python**. Opracowana metoda priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych (wraz ze wszystkimi modułami, modelami) została zaimplementowana w języku Python. Jest to wysokopoziomowy język programowania na licencji Open-Source, posiadający rozległą bazę ogólnodostępnych bibliotek (ang. package). Zostały wykorzystane, przede wszystkim, następujące biblioteki:

- **PyMathProg**

  Biblioteka Pythona służąca do modelowania, rozwiązywania i analizowania problemów programowania liniowego. Biblioteka korzysta z solvera GLPK (GNU Linear Programming Kit). GLPK wykorzystuje metodę Simplex. Biblioteka PyMathProg pozwala implementować zagadnienia programowania liniowego, kod jest przejrzysty oraz łatwy do zrozumienia.

- **NumPy (Python for Win32)**

  Biblioteka Pythona, która pozwala wykonywać obliczenia na wielowymiarowych macierzach. Wspomaga też obliczenia matematyczne oraz generowanie liczb losowych (przydatnych w opracowanej modyfikacji metody MCTS).

- **Pandas**

  Biblioteka Pythona wspomagająca analizę i zarządzanie danymi, przydatna do eksportowania danych z programu do np. Excela lub pliku CSV.

- **Plotly**

  Biblioteka Pythona, która pozwala w łatwy sposób tworzyć zaawansowane wykresy. Została wykorzystana przy generowaniu wykresów Gantta.

2. Uzasadnienie tematu rozprawy

2.1. Wstępne badania kwestionariuszowe

W celu zidentyfikowania problemów, jakie mogą wystąpić podczas realizacji przedsięwzięć wieloobiektowych, przeprowadzone zostały wywiady swobodne z osobami związanymi z projektowaniem realizacji lub realizacją tego typu przedsięwzięć. Wywiady przeprowadzono dwuetapowo. W pierwszym etapie rozmówcami byli przede wszystkim inżynierowie i kierownicy robót zajmujący się planowaniem i realizacją przedsięwzięć wieloobiektowych. W drugim etapie przeprowadzono wywiady z prezesami przedsiębiorstw zajmujących się planowaniem i realizacją przedsięwzięć wieloobiektowych. Respondenci brali udział w realizacji takich inwestycji jak: osiedla domów wielorodzinnych, osiedla bloków mieszkalnych oraz obiekty użyteczności publicznej (składające się z wielu obiektów o zbliżonej technologii).

W pierwszym etapie wywiady zostały przeprowadzone z pięcioma ekspertami (inżynierami i kierownikami). Podczas wywiadów eksperci odpowiadali na przygotowane pytania z zakresu problemów przy realizacji i harmonogramowaniu przedsięwzięć wieloobiektowych. W wywiadach pytano przede wszystkim o:

- świadomość wpływu kolejności wykonania obiektów na czas i koszt wśród inwestorów, generalnych wykonawców i inżynierów;
- wpływ inżynierów i kierowników na możliwość zmiany kolejności wykonania obiektów;
- ograniczenia kolejności wykonania obiektów narzuconych przez inwestora;
- analizę przez inżynierów różnych scenariuszy (kolejności) wykonania obiektów.

Dodatkowo respondenci opisywali swoje doświadczenia zdobyte podczas realizacji przedsięwzięć wieloobiektowych.

Respondenci nie mieli świadomości, że wybór kolejności wykonania obiektów ma wpływ na czas i koszty realizacji przedsięwzięć wieloobiektowych. Zwracali oni uwagę jedynie na problemy organizacyjne, wynikające z niewłaściwego wyboru kolejności realizacji obiektów. Problemy te dotyczyły między innymi rozmieszczenia żurawi wieżowych oraz wyznaczenia i wykonania dróg tymczasowych na placu budowy. Generalni wykonawcy stawiani są zazwyczaj w sytuacji, w której to inwestor określa kolejność wykonania obiektów poprzez narzucenie terminów dyrektywnych wykonania budynków. Jest to sytuacja niekorzystna zarówno dla wykonawcy, jak i inwestora. Inwestorzy lub deweloperzy niemający wiedzy na temat organizacji budowy, określają kolejność wykonywanych obiektów jedynie na podstawie przypuszczeń lub

opinii działu marketingu, który zaleca budowanie w pierwszej kolejności obiektów od wskazanej strony. Takie podejście ma zachęcić potencjalnych klientów do zakupu mieszkań. Najczęściej z konsekwencjami nieodpowiednio zaplanowanych robót zmagają się użytkownicy nowopowstałych obiektów od strony ulicy, gdyż samochody wjeżdżające na teren budowy do obiektów położonych w głębi, mogą spowodować zniszczenie dróg lokalnych. Generalny wykonawca w związku z nieprzemyślaną kolejnością realizacji obiektów boryka się z problemami organizacyjnymi oraz ponosi dodatkowe koszty, natomiast inwestor zostaje narażony na niedotrzymanie terminów przez wykonawcę. Generalni wykonawcy, mający wpływ na kolejność wyboru obiektów, kierując się zapewnieniem prawidłowej organizacji budowy, w pierwszej kolejności uwzględniają usytuowanie żurawi oraz dróg tymczasowych.

Istotnym problemem jest organizacja dostaw materiałów na teren budowy, szczególnie w przypadku realizacji składających się z kilkudziesięciu obiektów. Jeden z respondentów przytoczył przykład realizacji, podczas której ponad 60 samochodów ciężarowych i betonomieszarek samochodowych każdego ranka blokowało całkowicie plac budowy i okoliczne drogi lokalne. Przygotowanie i zapewnienie odpowiedniej obsługi dostaw przy tak dużych przedsięwzięciach jest niezwykle istotne i zarazem trudne logistycznie.

Kierownicy budów zwrócili uwagę, że istnieje możliwość skracania czasu wykonywania niektórych prac. Przyspieszenie czasu realizacji robót można uzyskać poprzez przeznaczenie dodatkowych środków na realizację przedsięwzięcia (pracowników, sprzętu), jednak wiąże się z ponoszeniem dodatkowych kosztów. Takie działanie jest jednak konieczne, aby uniknąć kar za niedotrzymanie terminów lub otworzyć nowe fronty dla pozostałych brygad roboczych.

W drugim etapie przeprowadzono wywiady swobodne z osobami decyzyjnymi (ekspertami) w dużych przedsiębiorstwach związanych z realizacją budowlanych przedsięwzięć wieloobiektowych. Nawiązano kontakt z Panem Ryszardem Trykosko (Wiceprezesem ds. Rozwoju Biznesu NDI), Panem Dariuszem Blocherem (aktualnym Członkiem Rady Nadzorczej, byłym wieloletnim Prezesem Zarządu Budimex S.A.) oraz z Panem Marcinem Wagą (Senior Project Managerem w przedsiębiorstwie Gleeds Polska Sp. z o.o.).

W wywiadach pytano o te same aspekty, o które pytano inżynierów i kierowników oraz dodatkowo o trend dotyczący liczby realizowanych przedsięwzięć wieloobiektowych. Eksperci twierdzą, że inżynierowie mają świadomość wpływu kolejności realizacji na czas i koszt. To stanowisko nie pokrywa się z odpowiedziami udzielonymi przez inżynierów. Eksperci zwrócili uwagę na mnogość czynników, które mogą warunkować kolejność wykonania obiektów. Wskazali takie czynniki jak:

- **Decyzje inwestora**

  Ze względów komercyjnych inwestor decyduje o realizacji poszczególnych obiektów. W pierwszej kolejności realizowane są obiekty, które mają już swoich przyszłych użytkowników i inwestor podpisał już umowy najmu lub umowę z operatorem na przykład hotelu.

- **Prawne**

  Niektóre fazy realizacji przedsięwzięcia mają prawomocne pozwolenie na budowę, a inne go nie posiadają.

- **Logistyczne**

  Rozpoczęcie realizacji przedsięwzięcia od obiektu, do którego jest najbardziej utrudniony dostęp, kwestie związane z zagospodarowaniem placu budowy lub dostęp do istniejącej infrastruktury (drogi oraz przyłącza).

- **Społeczne**

  Osoby zamieszkujące sąsiedztwo placu budowy mogą skutecznie blokować uzyskanie pozwolenia na budowę, co ma bezpośredni wpływ na kolejność realizacji obiektów.

- **Środowiskowe**

  Konieczność realizacji robót remediacyjnych może wpłynąć na kolejność realizacji poszczególnych obiektów.

- **Techniczne**

  Rozpoczęcie realizacji od obiektów najniżej posadowionych lub najmniej skomplikowanych.

- **Ekonomiczne**

  Sytuacja na rynku budowlanym, koniunktura, wzrost cen robocizny i materiałów budowlanych mogą wpłynąć na plany inwestycyjne inwestora, co z kolei wpłynie na kolejność wykonania obiektów.

Eksperci wskazują, że realizacje wieloobiektowe są powszechne, trudne i złożone, a więc inżynier na etapie przygotowania powinien analizować różne scenariusze realizacji. Eksperci prognozują też, że liczba tego typu realizacji będzie rosła, jednak jest to uwarunkowane wieloma czynnikami, takimi jak dostępność działek, sytuacja gospodarcza, spadek koniunktury, popyt.

Ważnym problemem są braki w oprogramowaniu w zakresie harmonogramowania przedsięwzięć wieloobiektowych. Do programów wspierających harmonogramowanie takich przedsięwzięć należy program KASS (Krzemiński Algorithm Scheduling System)(Krzemiński, 2016b) (Krzemiński, 2016c). Jest to program do harmonogramowania uwzględniający różne

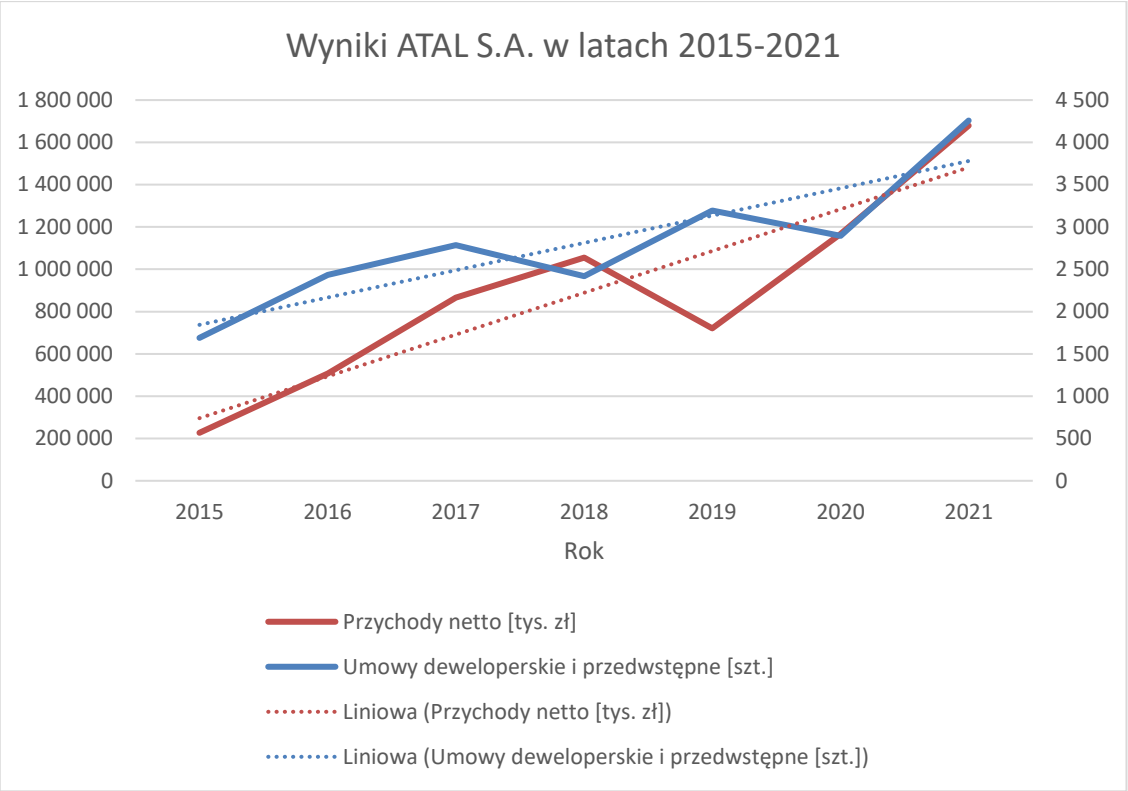
kryteria (czas, ciągłość pracy brygady, czas przejścia brygad pomiędzy obiektami). Ogólnodostępne programy (takie jak MS Project, Primavera, Planista) nie umożliwiają analizy różnych uszeregowań wykonywanych obiektów w prosty sposób. Dla inwestycji składającej się już z kilku obiektów praca w takim programie jest żmudna i nieefektywna. Nie ma oprogramowania wspomagającego wybór kolejności wykonywania obiektów przedsięwzięcia wieloobiektowego, biorącego pod uwagę jednocześnie koszt i czas. Istnieje konieczność opracowania programu uwzględniającego ograniczenia w sposób elastyczny.

2.2. Analiza rynku wieloobiektowych przedsięwzięć budowlanych

W tym rozdziale przedstawiono wyniki finansowe oraz liczbę sprzedawanych lokali przez trzy duże przedsiębiorstwa deweloperskie. Oczywiście nie wszystkie działania prezentowanych przedsiębiorstw to realizacja przedsięwzięć wieloobiektowych, ale jest ona znacząca część ich działalności, co wykazano poprzez zaprezentowanie ostatnich realizacji wieloobiektowych tych przedsiębiorstw. Dodatkowo przedstawiono też dane udostępniane przez Główny Urząd Statystyczny (GUS) na temat liczby mieszkań oddanych do użytku w latach 2015-2021. Nie wszystkie lokale oddane do użytku powstają w wyniku realizacji przedsięwzięć wieloobiektowych, jednak osiedla i zespoły budynków mieszkalnych mają na tę liczbę znaczący wpływ.

Na rysunku 2 przedstawiono wyniki ATAL S.A. Jak można przeczytać na stronie internetowej: „ATAL S.A to przedsiębiorstwo deweloperskie specjalizująca się w budownictwie kompleksów mieszkaniowych, apartamentów oraz lokali komercyjnych, zlokalizowanych w obrębie największych miast w Polsce”[1]. Jest to więc przedsiębiorstwo specjalizująca się w realizacji przedsięwzięć wieloobiektowych. Na wykresie można odczytać liczbę umów deweloperskich i przedwstępnych (Informacja o sprzedaży mieszkań przez ATAL S.A. w latach 2015-2021) oraz przychody netto ATAL S.A. Wyniki finansowe ATAL S.A.) na przestrzeni lat 2015-2021. Dodatkowo na wykresie został zaprezentowany również trend liniowy dla wspomnianych danych. Jak widać, przedsiębiorstwo wyspecjalizowane w realizacji przedsięwzięć wieloobiektowych notuje wzrost w latach 2015-2021 zarówno w liczbie zawartych umów (sprzedanych mieszkań), jak i przychodach. Na rysunku 3 przedstawiono wizualizację osiedla Skwer Harmonia, realizowanego przez ATAL S.A. w Krakowie. Realizacja składa się z 16 obiektów, a zakończenie prac przewidziano na 2024 r.

[1] https://atal.pl/atal-group/o-firmie - data dostępu 14.07.2022 r.



Rysunek 2: Wyniki ATAL S.A. w latach 2015-2021. Opracowanie własne. Dane: ATAL S.A.

Rysunek 3: Wizualizacja osiedla Skwer Harmonia, źródło: https://skwerharmonia.pl/galeria-wizualizacje-osiedla (data dostępu 18.03.2023 r.)

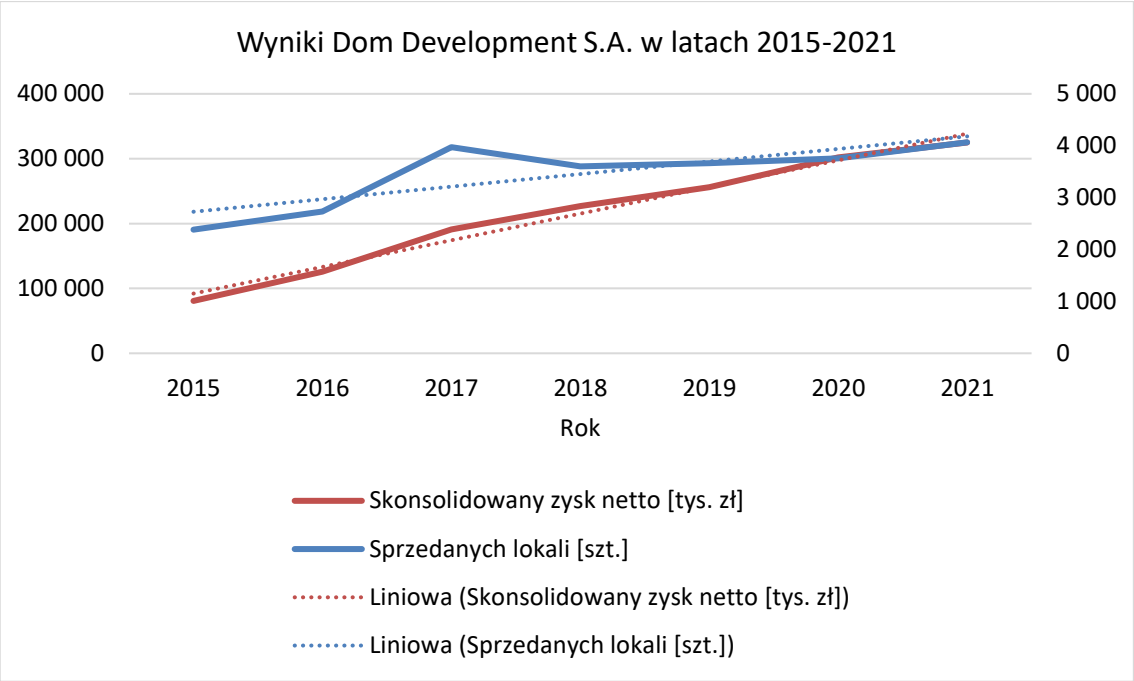
Inne najnowsze, wybrane inwestycje wieloobiektowe przedsiębiorstwa ATAL S.A. to[2]:

- Naramowice Odnova, Poznań, 4 obiekty, zakończenie prac: 2025 r.;
- Sokolska Towers, Katowice, 2 obiekty, zakończenie prac: 2019 r.;
- Apartamenty Ostródzka II, Warszawa, 4 obiekty, zakończenie prac: 2024 r.;
- Atal Olimpijska, Katowice, 3 obiekty, zakończenie prac: 2025 r.;
- Zacisze Marcelin II, Poznań, 6 obiektów, zakończenie prac: 2024 r.;
- Nowe Miasto Jagodno V, Wrocław, 4 obiekty, zakończenie prac: 2023 r.

Kolejnym przedsiębiorstwem, dla którego dokonano analizy wyników jest Dom Develepment, trzecie największe przedsiębiorstwo notowane na Giełdzie Papierów Wartościowych w Warszawie w sektorze nieruchomości (na dzień 28.11.2022 r.). Według informacji podanych na stronie internetowej: „Jesteśmy największym i jednym z najdłużej działających deweloperów na polskim rynku. Działamy od 1996 roku, a od 2006 roku jesteśmy notowani na Giełdzie Papierów Wartościowych w Warszawie. Do końca 2021 roku oddaliśmy do użytku 43 tysięcy mieszkań i apartamentów"[3]. Na rysunku 4 przedstawiono wykres prezentujący wyniki przedsiębiorstwa Dom Development w latach 2015-2021. Na wykresie przedstawiono liczbę sprzedanych lokali oraz skonsolidowany zysk netto (Sprawozdania finansowe za lata 2015-2021 Dom Development S.A.). Dodatkowo na wykres naniesiono trend liniowy dla obu danych. Tak jak w przypadku przedsiębiorstwa ATAL S.A., widać znaczące wzrosty zarówno w przypadku liczby sprzedanych lokali, jak i zysku. Na rysunku 5 przedstawiono plan Osiedla Wilno, realizowanego przez Dom Development. Osiedle, składające się z kilkudziesięciu obiektów, zamieszkuje ponad 10000 mieszkańców. Inne najnowsze, wybrane inwestycje wieloobiektowe przedsiębiorstwa Dom Develepment to[4]:

- Apartamenty Białej Koniczyny, Warszawa, 6 obiektów, zakończenie prac: 2024 r.;
- Osiedle Jagiellońska, Warszawa, 2 obiekty, zakończenie prac: 2024 r.;
- Apartamenty Solipska, Warszawa, 4 obiekty, zakończenie prac: 2024 r.;
- Dzielnica Mieszkaniowa Metro Zachód, Warszawa, 14 obiektów, zakończenie prac: 2023 r.;
- Górka Narodowa, Kraków, 23 obiekty, zakończenie prac: 2023 r.

[2] https://atal.pl/ - data dostępu 7.03.2023 r.
[3] https://www.domd.pl/pl-pl/krakow/o-nas - dostęp 28.11.2022 r.
[4] https://www.domd.pl/ - data dostępu 7.03.2023 r.



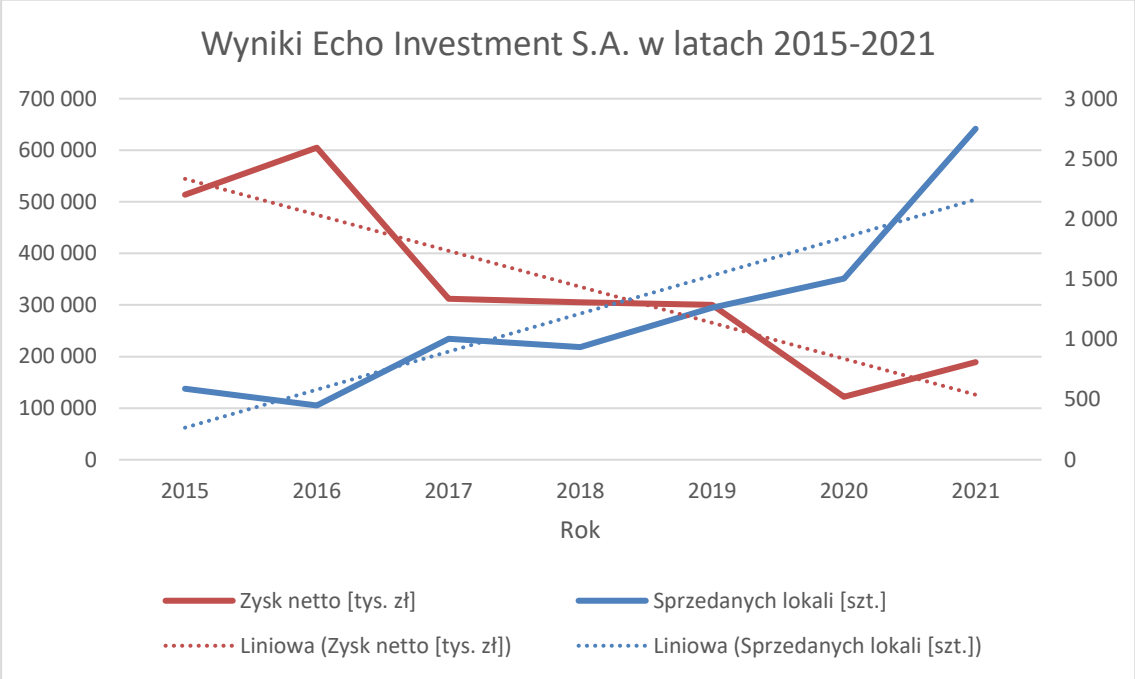
Rysunek 4: Wyniki Dom Development S.A. w latach 2015-2021. Opracowanie własne. Dane: Dom Development S.A.

Rysunek 5: Plan inwestycji Osiedle Wilno. Źródło: https://www.domd.pl/pl-pl/warszawa/lista-inwestycji/osiedle-wilno?city=warszawa (data dostępu 18.03.2023 r.)

Na rysunku 6 przedstawiono wykres prezentujący wyniki przedsiębiorstwa Echo Investment S.A. w latach 2015-2021. Na wykresie przedstawiono liczbę sprzedanych lokali oraz zysk netto (*Sprawozdania Zarządu z działalności spółki Echo Investment S.A. za lata 2015-2021*). Na wykres naniesiono linie trendu dla obu danych. Można zauważyć dość dynamiczny wzrost liczby sprzedanych lokali w poszczególnych latach. Od 2017 roku finansowe wyniki spółki notują

spadki. Jest to spowodowane zmianą modelu biznesowego, jaki przeprowadziła spółka w roku 2016 stawiając wyłącznie na działalność deweloperską. Jednak pomimo spadku zysków, można zauważyć stale rosnące zainteresowanie mieszkaniami oferowanymi przez spółkę Echo Investment. Najnowsze wybrane inwestycje wieloobiektowe przedsiębiorstwa Echo Investment to[5]:

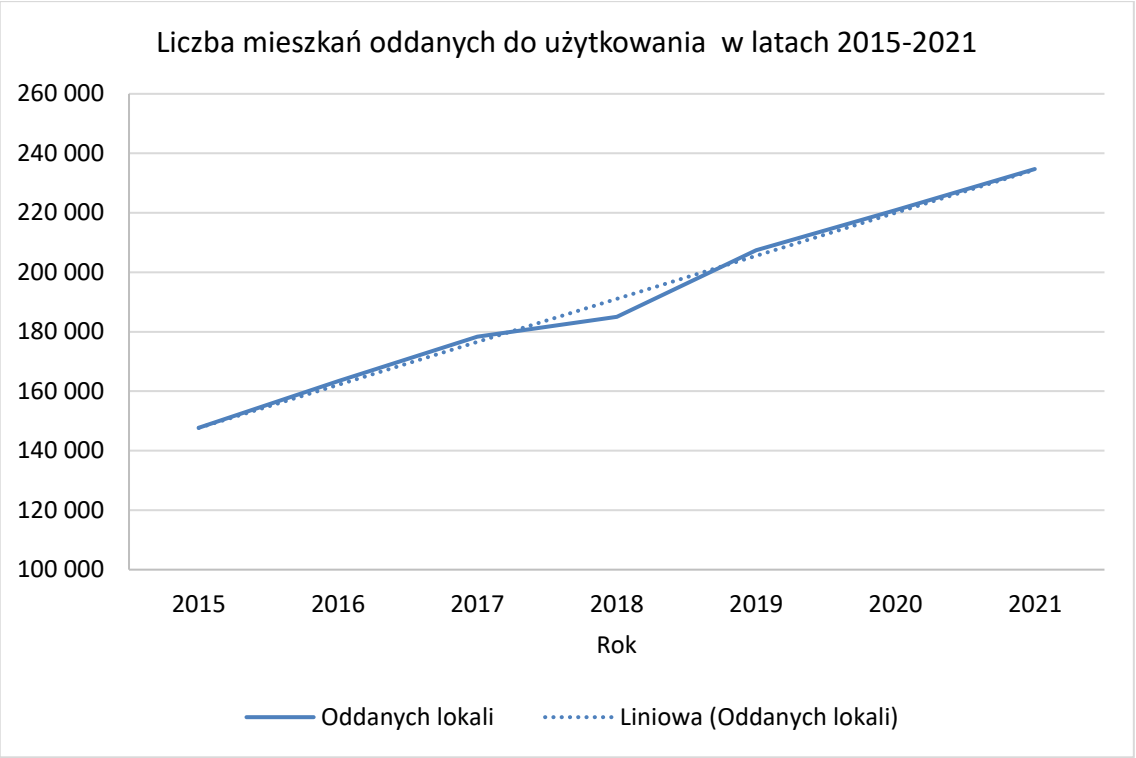
- Rytm, Warszawa, 6 obiektów, zakończenie prac: 2023 r.;
- Boho, Łódź, 2 obiekty, zakończenie prac: 2023 r.;
- Zam, Kraków, 3 obiekty, zakończenie prac: 2024 r.;
- Wieża Jeżyce, Poznań, 9 obiektów, zakończenie prac: 2023 r.



Rysunek 6: Wyniki Echo Investment S.A. w latach 2015-2021. Opracowanie własne. Dane: Echo Investment S.A.

Na rysunku 7 zostały przedstawione dane Głównego Urzędu Statystycznego na temat oddanych do użytku mieszkań w latach 2015-2021 (*Budownictwo mieszkaniowe w okresie styczeń-grudzień 2021 roku*, 2022). Na wykresie przedstawiono też linię trendu. Jak widać, liczba mieszkań oddawanych do użytku od 2015 roku cały czas rośnie. W 2015 roku było to 147 711 lokali, a w 2021 już 234 718. Wzrost liczby sprzedanych mieszkań wynosi rok do roku ok. 14 500 sztuk. Warto również zwrócić uwagę, że w roku 2021 ok. 60% mieszkań było wybudowanych przez deweloperów, a tylko 38% przez podmioty indywidualne (pozostałe na cele spółdzielcze, komunalne i inne).

[5] https://echo-mieszkania.pl/ - data dostępu 7.03.2023 r.



Rysunek 7: Liczba mieszkań oddanych do użytkowania w latach 2015-2021. Opracowanie własne. Dane: GUS.

2.3. Przegląd literatury polskiej i zagranicznej

Istnieje wiele metod i technik, które służą do harmonogramowania budowlanych przedsięwzięć wieloobiektowych. Mnogość metod powoduje nieporządek w nomenklaturze, dlatego w pierwszej kolejności podjęto próbę zestawienia najczęściej używanej nomenklatury w ramach harmonogramowania przedsięwzięć wieloobiektowych. W związku z faktem, że niektóre opracowane metody harmonogramowania przedsięwzięć wieloobiektowych trudno sklasyfikować, przedstawiono ogólne kierunki rozwoju tych metod. Następnie opisane zostały artykuły przy uwzględnieniu, czy występuje w nich tylko analiza czasowa, czy też kosztowa. Na końcu podrozdziałów 2.3.3 oraz 2.3.4 przedstawiono tabele, w których zaprezentowano zestawienie omawianych pozycji literaturowych (tabela 2: Wybrane publikacje uwzględniające analizę czasową w porządku chronologicznym oraz tabela 3: Wybrane publikacje uwzględniające analizę kosztową w porządku chronologicznym). Następnie przeanalizowano ogólnodostępne prace doktorskie i monografie w ramach harmonogramowania przedsięwzięć wieloobiektowych. Przeprowadzono również analizę liczby publikacji w bazie Scopus.

2.3.1. Nomenklatura występująca w harmonogramowaniu przedsięwzięć wieloobiektowych

Harmonogramowanie przedsięwzięć wieloobiektowych jest wieloaspektowym, multidyscyplinarnym i złożonym zagadnieniem. Zagadnieniem, które było analizowane z zastosowaniem wielu różnych metod, takich jak: teoria grafów, metoda ścieżki krytycznej, metoda pracy potokowej, metody szeregowania zadań, metoda sprzężeń czasowych i inne. Metody te zostaną pokrótce omówione w kolejnych podrozdziałach, jednak należy zwrócić uwagę na występujące problemy z nomenklaturą. W zależności od przyjętej metody, te same pojęcia mogą mieć różne nazwy, jak również te same nazwy potrafią określać różne pojęcia. Ze względu na nieporządek, jaki panuje w zakresie harmonogramowania przedsięwzięć wieloobiektowych (i harmonogramowania w ogóle), autor postanowił zestawić nazewnictwo spotykane w literaturze z podziałem na metody. Całość została przedstawiona w tabeli 1. W każdej kolumnie (oprócz pierwszej, która nazywa daną metodę) przedstawiono te same (lub zbliżone) pojęcia. Należy zwrócić uwagę, że nie każde pojęcie ma swoje odpowiedniki w różnych metodach. Wynika to z faktu, że niektóre metody są metodami ogólnymi harmonogramowania (np. metoda ścieżki krytycznej), a niektóre metodami szczegółowymi harmonogramowania (np. metoda sprzężeń czasowych).

W tabeli przedstawiono również zagraniczną nomenklaturę występującą w obszarze harmonogramowania przedsięwzięć wieloobiektowych oraz powtarzalnych. Przedstawiono najpopularniejsze nazewnictwo używane w metodach: LOB (*Line of Balance Technology: A Graphic Method of Industrial Programming*, 1962a), Repetitive Scheduling Method (Harris & Ioannou, 1998) oraz Repetitive Construction Projects (Zhang & Zou, 2015).

Zestawienie nie wyczerpuje całkowicie nazewnictwa w ramach harmonogramowania przedsięwzięć wieloobiektowych. Autor każdej metody wypracował swoje nazewnictwo, które może nieznacznie się różnić od pozostałych.

W dysertacji przyjęto nomenklaturę, którą wprowadziła metoda pracy potokowej, ponieważ zdaniem autora najlepiej oddaje specyfikę problemu harmonogramowania przedsięwzięć wieloobiektowych. Jednak w związku z inspiracją autora metodą sprzężeń czasowych, część pojęć będzie stosowana wymiennie, zwłaszcza jeżeli mowa o pojęciu „sprzężenia czasowe” oraz „założenia opisujące wymagania organizacyjne”. Poszczególne definicje pojęć zostały podane w podrozdziale 3.1. Opracowane na potrzeby pracy pojęcia takie jak podejście elastyczne oraz koncepcja harmonogramowania priorytetowego zostały wytłumaczone w rozdziale 3.2.

Tabela 1: Nomenklatura w harmonogramowaniu przedsięwzięć wieloobiektowych

Nazwa metody	Pojęcia						
Teoria grafów	Węzeł	Łuk	-	-	-	-	-
Metoda ścieżki krytycznej, Sieć jednopunktowa, Połoński	Zadanie	Powiązania technologiczne i organizacyjne	-	-	-	-	-
Metoda ścieżki krytycznej, Sieć dwupunktowa, Kelley, Walker	Zdarzenie	Zadanie	-	-	-	-	-
Szeregowanie zadań, Smutnicki	Operacja	-	Maszyna	Zbiór różnych typów maszyn	Zadanie	Zbiór zadań	Ciąg operacji konieczny do wykonania zadania
Metoda sprzężeń czasowych, Hejducki	Robota	Sprzężenie czasowe	Grupa robocza	Zbiór grup roboczych	Działka robocza	Kompleks działek roboczych	Zbiór robót konieczny do wykonania na działce roboczej
Metoda pracy potokowej, Marcinkowski	Proces roboczy	Założenia opisujące wymagania organizacyjne	Specjalistyczne zespoły, specjalistyczne brygady	Zbiór specjalistycznych zespołów, zbiór specjalistycznych brygad	Obiekt, front roboczy, działka robocza	Zbiór obiektów, zbiór frontów roboczych, zbiór działek roboczych	Zbiór procesów roboczych
Line of Balance, US Navy Department	Operation, activity	Sequence	Team, Crew	Teams, Crews	-	-	-
Repetitive Construction Projects, Li-hui Zhang i Xin Zou	Sub-activity	Work sequence between two sub-activities	Resource	Resources	Unit	Repetitive Construction Projects	-
Repetitive Scheduling Method, Harris I Ioannou	Activity	Sequence of activities	Resource	Resources	Unit	Multi-unit projects	-

2.3.2. Główne kierunki rozwoju metod harmonogramowania wieloobiektowych przedsięwzięć budowlanych

Przegląd literatury na temat harmonogramowania należy zacząć od wzmianki o Karolu Adamieckim, polskim inżynierze i ekonomiście, który jako pierwszy wprowadził metodę harmonogramowania. Pierwszy harmonogram powstał w 1896 roku (Marsh, 1975), natomiast pierwsze prace na temat harmonogramowania powstawały na początku XX wieku i wprowadzały innowacyjne jak na tamte czasy metody organizacji pracy (Adamiecki, 1909, 1931, 1938). W roku 1910 Henry Gantt opublikuje pracę, która rozpropagowała tworzenie harmonogramów na całym świecie i nadała jej graficznej formie nazwę „wykres Gantta" (Gantt, 1910).

Metoda Critical Path Method (CPM) opracowana w latach 50. XX wieku (Kelley & Walker, 1989) to najbardziej ogólna metoda harmonogramowania różnego typu przedsięwzięć – nie tylko wieloobiektowych, a nawet nie tylko budowlanych. Stosowana z sukcesami w programach do harmonogramowania i najbardziej rozpowszechniona wśród inżynierów. Metoda CPM polega na modelowaniu projektu jako sieci zależności między poszczególnymi zadaniami. W taki sposób określa się krytyczną ścieżkę projektu, czyli sekwencję zadań, której opóźnienie spowoduje opóźnienie całego projektu. Dzięki metodzie CPM osoby zarządzające projektami są w stanie dokładnie oszacować czas i koszty realizacji projektu, a także monitorować postępy prac i reagować na ewentualne opóźnienia (Kelley & Walker, 1989).

Technika linii równowagi (Line of Balance – LOB) powstała do zarządzania wieloma powtarzalnymi zadaniami w celu realizacji kilku takich samych projektów, zgodnie z przyjętym harmonogramem. Technika ta powstała w 1941 roku w przedsiębiorstwie Goodyear. Od połowy XX wieku była wykorzystywana w zarządzaniu projektami powtarzalnymi i niepowtarzalnymi przez marynarkę wojenną USA (*Line of Balance Technology: A Graphic Method of Industrial Programming*, 1962b). Technika ta realizowana jest w dwóch fazach:

- Faza 1: analiza wykonania pojedynczych projektów w serii projektów powtarzalnych;
- Faza 2: analiza wykonania serii projektów powtarzalnych zgodnie z kontraktem.

Wykorzystanie tej techniki dopełnia pozostałe metody planistyczne (harmonogramy), a nawet wstępne kosztorysowanie. Może też służyć jako narzędzie do śledzenia postępów przy realizacji projektów. Wadą tej techniki jest przede wszystkim mała skuteczność w przypadku skomplikowanych projektów (Bukłaha, 2016).

Istnieje wiele możliwości zastosowanie LOB w planowaniu inwestycji wieloobiektowych. Jest ona wykorzystywana w:

- harmonogramowaniu autostrad, długich mostów, rurociągów (Hegazy, 2001);
- wykonywaniu budynków wielopiętrowych (Mendes Jr i in., 1998);
- planowaniu pracy biura projektowego (Al Sarraj, 1990).

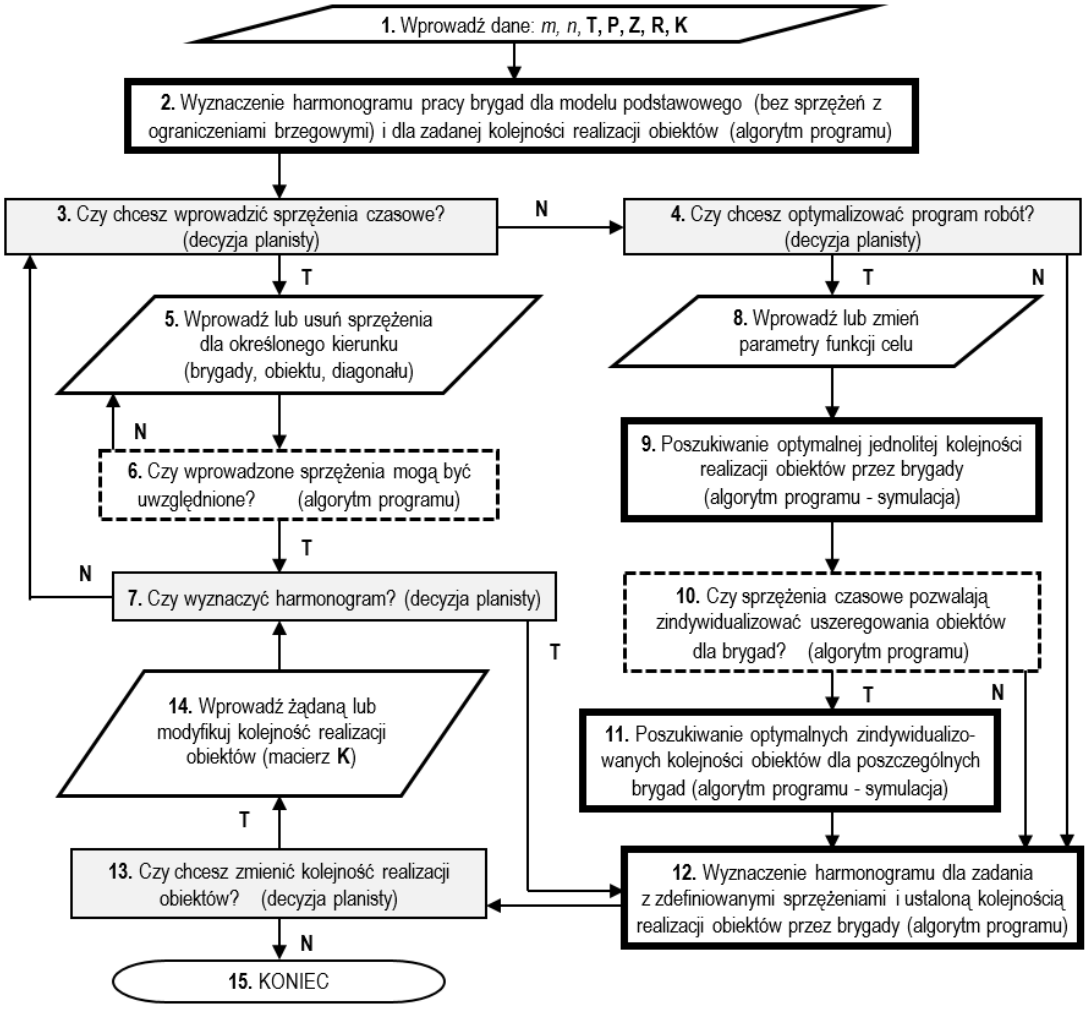
Metoda LOB została bardzo rozwinięta i powstało wiele jej modyfikacji, takich jak:

- Time Space Scheduling Method (Stradal & Cacha, 1982);
- Time-Location Matrix Model (Birrell, 1980);
- Vertical Production Method (O'Brien, 1975);
- Horizontal and Vertical Logic Scheduling for Multistory Projects (HVLS) (Thabet & Beliveau, 1994);
- Repetitive Project Model (RPM) (Reda, 1990);
- i inne.

Kolejną grupą metod przydatnych przy harmonogramowaniu przedsięwzięć powtarzalnych jest Metoda Sprzężeń Czasowych (Time Couplings Method – TCM). Metoda ta jest intensywnie rozwijana w Polsce. Stanowiła ona punkt wyjścia do stworzenia opracowanej przez autora koncepcji priorytetowego harmonogramowania. Metoda Sprzężeń Czasowych zaproponowana została przez prof. Afanaseva pod koniec lat siedemdziesiątych XX wieku. Przykładowe prace, w których przedstawiono podstawy teoretyczne metody sprzężeń czasowych oraz różne jej warianty to (Afanasev, 1977), (Afanasev, 1980) oraz (Mrozowicz, 1997). Metoda TCM jest ciągle rozwijana (Rogalska & Hejducki, 2022), (Kostrzewa & Rogalska, 2019), (Rogalska & Hejducki, 2017), (Hejducki & Rogalska, 2011) również w kierunku teorii ograniczeń (Goldratt, 1990) oraz metody łańcucha krytycznego (Goldratt, 1997), (Rogalska & Hejducki, 2007), (Rogalska & Hejducki, 2004).

Należy jeszcze wspomnieć o innym kierunku rozwoju harmonogramowania przedsięwzięć budowlanych. W artykule (Marcinkowski, 2017) przedstawiono metodę harmonogramowania interaktywnego. Koncepcja interaktywnego sposobu harmonogramowania zakłada, że algorytm metody optymalizacji nie będzie zależał od wprowadzanych na poszczególnych etapach obliczeń danych, które odwzorowują sytuacje planistyczną. Schemat modelu interaktywnego planowania pracy brygad w systemie pracy potokowej przedstawiono na rysunku 8. W interaktywnym modelu planowania to planista w kooperacji z programem ustala ograniczenia, jakie należy uwzględnić podczas optymalizacji. Ograniczenia są uwzględniane w postaci sprzężeń czasowych. Sprzężenia czasowe planista wprowadza iteracyjnie, a program za każdym razem sprawdza, czy

uwzględnienie nowego sprzężenia jest możliwe i odrzuca sprzężenia, które powodują, że obszar dopuszczalny problemu optymalizacyjnego będzie pusty. W algorytmie uwzględniono również możliwość optymalizacji kolejności wykonania obiektów. Metoda harmonogramowania interaktywnego była drugą, oprócz Metody Sprzężeń Czasowych, inspiracją autora do opracowania koncepcji priorytetowego harmonogramowania.



Rysunek 8: Schemat modelu interaktywnego planowania pracy w systemie pracy potokowej
Źródło: (Marcinkowski, Modelowanie ograniczeń w metodzie pracy potokowej, 2017)

2.3.3. Metody uwzględniające analizę czasową

W niniejszym podrozdziale przedstawiono wybrane pozycje literaturowe, w których zastosowano metody harmonogramowania przedsięwzięć wieloobiektowych uwzględniające analizę czasową.

W artykule (Rogalska i in., 2009) zaprezentowano podejście rozmyte w klasycznym problemie szeregowania zadań. Jako kryterium optymalizacyjne przyjęto całkowity czas trwania inwestycji. Ograniczeniami była ciągłość pracy brygad. Czas trwania poszczególnych procesów

opisują trzypunktowe liczby rozmyte. W celu optymalizacji harmonogramu zastosowano metodę Taboo Search (ATS). Autorzy zauważają, że przedstawianie czasów trwania prac jako liczby rozmyte wpływa na mniejszą wrażliwość na zmiany parametrów rozpatrywanego zagadnienia.

W publikacji (Bożejko i in., 2012) został przedstawiony problem planowania inwestycji wieloobiektowej z wykorzystaniem algorytmu memetycznego. Jest to algorytm hybrydowy, w którego skład wchodzi algorytm genetyczny oraz algorytm przeszukiwania lokalnego Taboo Search. Autorzy zastosowali również podejścia niedeterministyczne. Jednym z podejść było podejście probabilistyczne. Czas trwania każdej pracy został opisany jako niezależna zmienna losowa o rozkładzie normalnym. Autorzy zakładają arbitralną znajomość wartości oczekiwanej oraz odchylenia standardowego. Drugim podejściem było podejście rozmyte. Czas trwania każdej pracy został reprezentowany przez trzypunktowe liczby rozmyte. Autorzy podkreślają większą skuteczność podejścia probabilistycznego (mniejszy średni błąd względny).

W artykule (Bożejko i in., 2014) zaprezentowano metaheurystyczny algorytm poszukiwania minimalnego czasu trwania dla problemu szeregowania zadań z możliwością zazębiania się prac brygad na działkach roboczych. Jako funkcję celu przyjęto minimalny czas realizacji całej inwestycji. Autorzy zaproponowali algorytm przeszukiwania lokalnego taboo search. Został on przystosowany do ograniczeń i specyfiki problemu. Algorytm wykorzystano do planowania naprawy drogi. Remontowana droga została podzielona na 7 odcinków. Każdy odcinek jest utożsamiany z osobnym obiektem. Na poszczególnym odcinku należało wykonać 8 różnych prac. Autorzy artykułu przeprowadzili również test na danych wzorcowych (nawet dla 50 obiektów i 20 robót na obiekcie) z zastosowaniem ATS otrzymując wyniki średnio o ok. 3% lepsze niż uzyskane algorytmem MNEH.

W pracy (Jaśkowski & Biruk, 2014) autorzy proponują zastosowanie programowania mieszanego (liniowo-całkowitoliczbowego) do minimalizacji przestojów w pracy brygad roboczych. Model zakłada możliwość wyboru wariantu organizacji brygady roboczej. Każdy wariant charakteryzuje się różnymi czasami realizacji poszczególnych prac danej brygady na różnych działkach. Opracowane podejście zilustrowano na przykładzie przedsięwzięcia wieloobiektowego.

Kolejna praca prezentuje nurt popularnego w ostatnim czasie podejścia do projektowania odpornych harmonogramów. W pracy (Chen i in., 2014) zaproponowano wieloetapowy algorytm bazujący na przeszukiwaniu rojowym (algorytm ABC - Artificial Bee Colony). W pracy wzięto pod uwagę ograniczone zasoby. W modelu optymalizacyjnym wzięto pod uwagę funkcję entropii, która odzwierciedlała odporność harmonogramu i jego niezawodność.

W publikacji (Dolabi & Afshar, 2016) zostało zaproponowane ulepszenie metody LOB, polegające na możliwości zmiany tempa realizacji poszczególnych prac. Zmiana tempa może nastąpić przez zatrudnienie dodatkowych załóg lub ograniczenie pierwotnie przypisanych brygad. Do optymalizacji został zastosowany algorytm Max-Min Ant System (MMAS). Zaproponowany model charakteryzował się pozytywnym wpływem na rozwiązanie optymalne oraz większą elastycznością planowania produkcji w porównaniu z innymi modelami uwzględniającymi zmianę tempa produkcji.

W pracy (Krzemiński, 2016a) przedstawiono podejście probabilistyczne do metody szeregowania zadań. W artykule zaprezentowano sposób oceny stabilności harmonogramów budowlanych. Przyjęto trójkątne rozkłady prawdopodobieństwa czasu trwania pracy brygad na obiektach. Dla różnych uszeregowań obiektów dokonano symulacji, uzyskując przybliżony rozkład prawdopodobieństwa. Jako kryterium przyjęto prawdopodobieństwo dotrzymania terminu dyrektywnego. Autor zauważył, że uszeregowania, które posiadają większe zapasy czasu będą rozwiązaniami lepszymi - w rozumieniu - bardziej odpornymi na zakłócenia. Do przeprowadzania obliczeń został użyty program KASS (Krzemiński Algorithm Scheduling System).

W artykule (Radziszewska-Zielina & Sroka, 2016) podobnie jak w (Krzemiński, 2016a) zastosowano podejście probabilistyczne. Przyjęto, że czas trwania pracy brygady na obiekcie dany jest jednostajnym rozkładem prawdopodobieństwa. Do wyznaczenia rozkładu prawdopodobieństwa wykonania całej inwestycji zastosowano metodę analityczną opisaną w (Milian, 2006). Jako kryterium przyjęto termin wykonania inwestycji przy 95% prawdopodobieństwie. Przedstawiona metoda wymaga skomplikowanych obliczeń całkowych, zaleca się więc korzystanie z niej przy małych inwestycjach. Dla dużych inwestycji należy posłużyć się metodą symulacyjną.

W artykule (Podolski, 2016b) został przedstawiony model harmonogramowania przedsięwzięcia wieloobiektowego, w którym jest możliwość zastosowania więcej niż jednej grupy roboczej do wykonywania danej roboty w obiekcie oraz zależności kolejnościowe między robotami w danym obiekcie w formie sieci, jak w metodzie CPM. Zadanie optymalizacyjne minimalizacji czasu przedsięwzięcia zostało rozwiązane za pomocą autorskiego programu w systemie Mathematica, przy użyciu algorytmu Tabu Search.

W publikacji (Gouda i in., 2017) autorzy stosują hybrydowe podejście do planowania przedsięwzięć powtarzalnych. Zaprezentowany algorytm łączy metodę LOB z teorią grafów w celu uzyskania optymalnej alokacji zasobów wykorzystywanych podczas realizacji przedsięwzięcia. Zaletą modelu jest to, że wspiera on decydentów w formułowaniu optymalnej kolejności wykonywania robót przez brygady, a także w rozważeniu przydziału wykwalifikowanych załóg wielozadaniowych. Analiza studium przypadku wykazała, że

zastosowanie proponowanego modelu może znacznie zmniejszyć liczbę załóg pracujących podczas przedsięwzięć o charakterze powtarzalnym.

W pracy (Krzemiński, 2017) przedstawiono algorytm optymalizacji harmonogramów budowlanych uwzględniający potokową metodę organizacji pracy. Podstawowym założeniem jest wielozadaniowość brygad roboczych. Brygady mogą wspomagać przy wykonywaniu pracy inne brygady. Jeżeli brygada wspomaga inną brygadę, jej wydajność przy realizacji danego zadania jest mniejsza lub równa wydajności brygady wyspecjalizowanej do tego typu zadań. W procesie iteracyjnego poprawiania harmonogramu wyjściowego, algorytm pozwala usunąć przerwy w pracy brygad, a nawet skrócić czas trwania całej realizacji.

W pracy (Altuwaim & El-Rayes, 2018) został przedstawiony model harmonogramowania przedsięwzięć o charakterze powtarzalnym. Autorzy zaproponowani nowy algorytm minimalizujący zarówno czas trwania przedsięwzięcia, jak i przerwy w pracy brygad roboczych. Algorytm zakłada iteracyjne wyznaczane odpowiednich terminów (najwcześniejszych i najpóźniejszych) w kilku fazach. Model pozwala na wygenerowanie terminów najwcześniejszych i najpóźniejszych rozpoczęcia procesów, obliczenie dwóch nowatorskich wskaźników określających ciągłość pracy brygad, wyznaczenie szerokiego zakresu harmonogramów pomiędzy harmonogramami najwcześniejszymi i najpóźniejszymi oraz wyznaczenie harmonogramu, który najlepiej pasuje do konkretnego przedsięwzięcia.

W pracy (García-Nieves i in., 2018) została zaprezentowana metoda harmonogramowania przedsięwzięć o charakterze powtarzalnym przy ograniczonych zasobach. Autorzy proponują optymalizację ze względu na czas realizacji przedsięwzięcia oraz przerwy w pracy brygad roboczych. Opracowany matematyczny model został zweryfikowany na rzeczywistej realizacji składającej się z 4 wielopiętrowych budynków. Niewątpliwą zaletą proponowanej metody jest zaimplementowanie jej w Excelu, co może być przydatne dla użytkownika.

W publikacji (Tomar & Bansal, 2019) zaproponowano połączenie metod CPM oraz LOB w celu harmonogramowania przedsięwzięć budowlanych, podczas których realizowane są zarówno roboty powtarzalne jak i niepowtarzalne. Dodatkowo metody CPM oraz LOB zostały zintegrowane z systemem informacji geograficznej (GIS), co pozwala uwzględnić wpływ otoczenia na opracowany harmonogram. Zaproponowana przez autorów metoda pozwala połączyć zalety metod CPM (zależności logiczne pomiędzy robotami) oraz LOB (stabilność zużycia zasobów). Metoda została zaimplementowana oraz przetestowana na przykładzie praktycznym.

Autorzy publikacji (Biruk & Rzepecki, 2019) postulują o uwzględnienie efektu uczenia i zapominania (Learning-Forgetting) przy harmonogramowaniu robót budowlanych o charakterze powtarzalnym. Brygady, wykonując powtarzalny proces, zwiększają swoją wydajność, co

powoduje skrócenie czasu wykonania zadania, tym samym zmniejszenie kosztów jego wykonania. Przerwa w pracy brygad, spowodowana ograniczeniami technologiczno-organizacyjnymi, skutkuje spadkiem wydajności. Uwzględnienie efektu uczenia i zapominania może mieć kluczowy wpływ nie tylko na czas, ale też na koszt realizacji, a tym samym może stanowić przewagę na etapie składania ofert w przetargach.

W publikacji (Tomczak & Jaśkowski, 2022) zostało zaproponowane deterministyczne podejście do harmonizacji prac w przedsięwzięciach wieloobiektowych z powtarzalnymi niejednorodnymi procesami. W funkcji celu uwzględniono całkowity czas trwania projektu, czas realizacji poszczególnych obiektów oraz przestoje w pracy brygad. Ważność poszczególnych składowych w funkcji metakryterium określa planista. Do rozwiązania został zastosowany algorytm przeszukiwania rojem cząstek. Skuteczność działania modelu została sprawdzona na przykładzie budowy osiedla domów typu bliźniak.

W tabeli 2 przedstawiono zestawienie artykułów uwzględniających analizę czasową w porządku chronologicznym. Pojęcie uwzględnienia ograniczeń w sposób elastyczny został wytłumaczony w rozdziale 3.2.

Tabela 2: Wybrane publikacje uwzględniające analizę czasową w porządku chronologicznym

Autor, rok	Podejście/metoda	Zastosowanie	Ograniczenia metody
Rogalska, Bożejko, Hejducki i Wodecki, 2009	Rozmyte, algorytm Tabu Search	Praca teoretyczna	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Bożejko, Hejducki, Rajba i Wodecki, 2012	Deterministyczne, probabilistyczne, rozmyte, algorytm memetyczny	Harmonogramowanie przedsięwzięcia wieloobiektowego	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Bożejko, Hejducki, Uchroński i Wodecki, 2014	Deterministyczne, algorytm Tabu Search	Naprawa drogi podzielona na 7 odcinków	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Jaśkowski i Biruk, 2014	Deterministyczne, programowanie mieszane (liniowo-całkowitoliczbowe)	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Chen, Liang i Padilla, 2014	Deterministyczne, wieloetapowy algorytm bazujący na przeszukiwanie rojem pszczół	Harmonogramowanie przedsięwzięć przy ograniczonych zasobach	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Dolabi, 2016	Deterministyczne, ulepszona metoda linii równowagi, algorytm Max-Min Ant System (MMAS)	Harmonogramowanie przedsięwzięcia wieloobiektowego	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Krzemiński, 2016	Probabilistyczne, metoda symulacji	Prace teoretyczna	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów,

			trudność wyznaczenia czasów trwania w sposób probabilistyczny
Radziszewska-Zielina i Sroka, 2016	Probabilistyczne, rozwiązanie analityczne	Praca teoretyczna	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów, trudność wyznaczenia czasów trwania w sposób probabilistyczny
Gouda, Hosny i Nassar, 2017	Linia równowagi, Teoria grafów	Budowa rurociągu	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Krzemiński, 2017	Algorytm iteracyjny	Harmonogramowanie przedsięwzięcia wieloobiektowego	Delegowanie brygad do innych zadań nie zawsze jest możliwe, ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Podolski, 2017	Deterministyczne, Tabu Search	Harmonogramowanie przedsięwzięcia wieloobiektowego	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Altuwaima i El-Rayesa, 2018	Iteracyjne, wielofazowe wyznaczanie terminów	Harmonogramowanie przedsięwzięcia wieloobiektowego	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Garcia-Nieves, Ponz-Tienda, Salcedo-Bernal i Pellicer, 2018	Programowanie matematyczne	Harmonogramowanie przedsięwzięcia wieloobiektowego (4 budynków wielopiętrowych)	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Tomar i Bansal, 2019	Linia równowagi, metoda ścieżki krytycznej	Harmonogramowanie przedsięwzięcia powtarzalnego	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Biruk i Rzepecki, 2019	Deterministyczne, mechanizm uczenia i zapominania	Harmonogramowanie przedsięwzięcia powtarzalnego	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów
Tomczak i Jaśkowski, 2022	Deterministyczne, przeszukiwanie rojem cząstek	Harmonogramowanie przedsięwzięcia wieloobiektowego (osiedle domów typu bliźniak)	Ograniczenia uwzględnione w sposób nieelastyczny, brak uwzględnienia kosztów

2.3.4. Metody uwzględniające analizę kosztową

W niniejszym podrozdziale zaprezentowano wybrane pozycje literaturowe, w których przedstawiono projektowanie realizacji przedsięwzięć wieloobiektowych uwzględniające analizę kosztową.

Celem pracy (Moselhi & El-Rayes, 1993) jest przedstawienie modelu, który uwzględnia koszt jako ważną zmienną w procesie optymalizacji. Model ten umożliwia optymalny dobór wielkości brygad roboczych przy wykonywaniu przedsięwzięć wieloobiektowych. W modelu

zostało zastosowane dwuetapowe programowanie dynamiczne. W pierwszym etapie została wykorzystana analiza kompromisowa kosztów w celu odnajdowania rozwiązania suboptymalnego. W drugim etapie użyto prostego skanowania i selekcji w celu polepszenia rozwiązania suboptymalnego. Model został przetestowany na przykładzie literaturowym. Ci sami autorzy w publikacji (Moselhi & El-Rayes, 1993) uwzględnili wpływ pogody oraz efekt uczenia się brygad przy realizacji przedsięwzięć wieloobiektowych. W modelu ponownie wykorzystano programowanie dynamiczne do optymalizacji kosztów całkowitych przedsięwzięcia wieloobiektowego. W publikacji został udowodniony znaczący wpływ niekorzystnych warunków pogodowych oraz efekt uczenia się na całkowite koszty realizacji przedsięwzięcia wieloobiektowego.

W pracy (Hegazy & Wassef, 2001) przedstawiono model planowania i optymalizacji przedsięwzięć powtarzalnych. Model umożliwia minimalizację kosztów całkowitych, w skład których wchodzą koszty bezpośrednie, pośrednie, koszty przestojów oraz koszty likwidacji szkód. W założeniach modelu istnieje możliwość wyboru technologii oraz wielkości brygad. Do optymalizacji autorzy zastosowali algorytmy genetyczne. Model zaimplementowano w postaci łatwego w użyciu szablonu arkusza kalkulacyjnego. Model został przeliczony na przykładzie obliczeniowym.

W książce (Marcinkowski, 2002) została zaprezentowana metoda sterowania pracą specjalistycznych brygad roboczych przy planowaniu procesów budowlanych z zastosowaniem metod potokowych. Autor formułuje problem optymalizacyjny uwzględniający dyrektywne terminy dostępności brygad, terminy dyrektywne otwierania i zamykania frontów oraz przestoje w pracy brygad przy realizacji procesów na niejednorodnych frontach roboczych. Funkcja celu jest sumą kosztów, jakie należy ponieść z tytułu kar za niedotrzymanie terminu dyrektywnego wykonania obiektów oraz kosztów za przerwy w pracy brygad roboczych. Do poszukiwania rozwiązań suboptymalnych użyto zmodyfikowanej metody podziału i ograniczeń. Został również przedstawiony model dopuszczający różną kolejność realizacji prac przez brygady na frontach. W pracy położono nacisk na aspekt praktyczny omawianych modelów. Zwrócono jednak uwagę na jednostkową liczność brygad, a więc brak możliwości podziału brygady na grupy realizujące procesy na różnych frontach.

Publikacja (Senouci & Eldin, 2004) proponuje zastosowanie rozszerzonego algorytmu genetycznego Lagrange'a do rozwiązania problemu planowania zużycia zasobów. Dodatkowo w pracy została uwzględniona, poza typowymi relacjami pomiędzy robotami oraz ograniczeniami w zużyciu zasobów, wartość całkowitych kosztów projektu. Funkcja celu uwzględniała zarówno czas, całkowity koszt realizacji jak i dodatkowo ograniczenia uwzględnione w postaci kwadratowej funkcji kary.

W pracy (Mika i in., 2005) przedstawiono rozwiązanie problemu planowania projektów wielomodalnych przy ograniczonych zasobach i przy uwzględnieniu zdyskontowanych przepływów pieniężnych. Metoda bierze pod uwagę różne modele płatności. Zaproponowany model uwzględnia maksymalizację wartość netto wszystkich przepływów finansowych. W celu szukania suboptymalnych rozwiązań zastosowano metody symulowanego wyżarzania oraz przeszukiwanie z zabronieniami. Metoda została przetestowana na generatorze projektów z uwzględnieniem równomiernego rozkładu przepływów.

W pracy (Liu & Wang, 2008) zastosowano programowanie ograniczeń w celu minimalizacji kosztów przedsięwzięcia wieloobiektowego. Przedstawiony przez autorów model zakłada dwuetapową optymalizację. Najpierw dokonywana jest optymalizacja kosztów całkowitych przedsięwzięcia. Na koszty całkowite składają się koszty bezpośrednie (rozbite na koszty robocizny, materiałów i sprzętu) i pośrednie (zależność liniowa od czasu trwania przedsięwzięcia). Do wykonania każdego procesu można wybrać jedną z kilku dostępnych brygad. Po doborze brygad, które generują najmniejsze koszty całkowite, następuje optymalizacja minimalizująca przerwy w ich pracach. Model został zastosowany do wyznaczenia harmonogramu realizacji mostu.

W artykule (Rogalska i in., 2008) dokonano minimalizacji kosztów zasobów (pracowników) podczas realizacji przedsięwzięcia wieloobiektowego. Optymalizacja została wykonana algorytmem HEA (Hybrid Evolutionary Algorithm). Jako funkcję celu przyjęto odchylenie liczby pracowników od średniej podczas realizacji całego przedsięwzięcia. W drugim przykładzie uwzględniono wypłacanie pracownikom wynagrodzenia w transzach, w zależności od postępów przydzielonego zadania. Celem było możliwie jak najpóźniejsze wypłacanie wynagrodzenia przez pracodawcę. Założono również, że wykonanie zadania w ostatnim możliwym terminie wiąże się z dodatkowymi kosztami w wysokości 0%, 2,5%, 5%, 10%. Uzyskane wyniki porównano z algorytmem genetycznym. Proponowany algorytm HEA uzyskał o około 3,5% lepsze wyniki w porównaniu z algorytmem genetycznym.

W publikacji (San Cristóbal, 2009) został podjęty problem analizy czasu, kosztu i jakości przy wykonywaniu przedsięwzięć drogowych. Przedstawiany model zakłada możliwość minimalizacji czasu wykonania inwestycji przy założonym koszcie i jakości. Model można również zastosować do minimalizowania kosztu przy założonym czasie oraz jakości lub do maksymalizowania jakości przy założonym czasie i koszcie. W optymalizacji zastosowano programowanie binarne oraz algorytm genetyczny. Publikacja była odpowiedzią na wytyczne rządu USA w sprawie zapewniania najwyższej jakości usług budowlanych przy najniższych kosztach i najkrótszym czasie. Podobną analizę przeprowadzono w publikacji (El-Rayes & Kandil, 2005), jednak przedstawiona tam została optymalizacja wielokryterialna ze względu na koszt,

czas oraz jakość z wykorzystaniem algorytmów genetycznych. Zaprezentowany model umożliwia uzyskanie kompromisu pomiędzy czasem, kosztem oraz jakością dla przedsięwzięcia wieloobiektowego. Model może zostać zastosowany głównie do realizacji odcinków liniowych, takich jak autostrady.

W opracowaniu (Ezeldin & Soliman, 2009) podjęto problem optymalizacji czasowo-kosztowej przedsięwzięć wieloobiektowych. Do znalezienia rozwiązania optymalnego autorzy posłużyli się metodą hybrydową. Metoda ta była połączeniem algorytmów genetycznych oraz programowania dynamicznego. Algorytm genetyczny znajduje rozwiązanie przybliżone, natomiast programowanie dynamiczne wykorzystuje rozwiązanie przybliżone do odnalezienia globalnego rozwiązania optymalnego. Model uwzględnia czynniki, które wpływają na koszt i czas trwania zarówno na poziomie poszczególnych zadań, jak i na poziomie całego projektu. Studium przypadku przeprowadzono na przykładzie sieci stacji benzynowych. Z otrzymanych wyników można wyciągnąć wniosek, że algorytmy genetyczne mogą być zintegrowane z dynamicznymi technikami programowania, aby zapewnić skuteczne sposoby tworzenia optymalnych harmonogramów przedsięwzięć wieloobiektowych.

W artykule (Jaśkowski, 2015) autor zastosował programowanie liniowe całkowitoliczbowe (Mixed-Integer Linear Programming) do minimalizacji kosztów przy jednoczesnej minimalizacji przestojów w pracy brygad roboczych. Model zakłada możliwość wyboru wariantu organizacji brygady roboczej. Każdy wariant charakteryzuje się różnymi czasami oraz kosztami realizacji poszczególnych procesów. Głównym założeniem w opisywanym modelu jest możliwość wyboru rozwiązania technologiczno-organizacyjnego, które wpływa zarówno na koszty, jak i czas wykonania prac. W modelu nie przewidziano możliwości zmiany uszeregowania wykonywanych obiektów.

W artykule (Zhang i in., 2015) przedstawiono model czasowo-kosztowy, który zakładał możliwość zmiany kolejności wykonywania niektórych procesów. W prezentowanym modelu zostało zastosowane programowanie mieszane nieliniowe z wykorzystaniem logiki miękkiej (Soft Logic) oraz algorytmów genetycznych. Proponowany model został zastosowany do optymalizacji kosztów realizacji mostu. Z zaprezentowanego przykładu wynikało, że proponowana metoda zapewnia uzyskiwanie lepszych wyników oraz większą elastyczność przy realizacji przedsięwzięć powtarzalnych.

W artykule (Podolski, 2016a) zaproponowano model czasowo kosztowy. Zmiennymi decyzyjnymi w modelu optymalizacyjnym jest kolejność wykonywanych obiektów oraz sposób realizacji danego procesu. Każdy proces może być wykonany na 3 sposoby różniące się czasem i kosztem realizacji. Ograniczeniem jest dotrzymanie terminu dyrektywnego. Funkcja celu jest sumą kosztów wykonania wszystkich procesów. Jest to model optymalizacji dyskretnej

jednokryterialnej. Do rozwiązania tego problemu użyto algorytmu symulowanego wyżarzania. Przedstawiony model może znaleźć zastosowanie podczas ustalania optymalnego harmonogramu przedsięwzięcia wieloobiektowego przy wykorzystaniu potokowego systemu pracy.

W artykule (Tran i in., 2018) przedstawiono innowacyjną metodę optymalizacji wielokryterialnej uwzględniającej czas trwania, koszt, przerwy w pracy brygad oraz zasoby przy realizacji przedsięwzięć powtarzalnych. Proponowane podejście zostało nazwane "Oposition Multiple Objective Symbiotic Organisms Search" (OMOSOS). Zaprezentowana metoda opiera się o metody sztucznej inteligencji oraz heurystyk. Podejście OMOSOS może być łatwo przystosowane do różnych metod optymalizacji heurystycznej (jak przeszukiwanie rojem cząstek lub algorytmem mrówkowym). Na przykładach zaprezentowano odnajdywanie rozwiązania optymalnego w sensie Pareto dla dwóch przedsięwzięć powtarzalnych - jednym była budowa mostu betonowego, drugim prace ogólnobudowlane.

W pracy (Rosłon & Kulejewski, 2019) zostało zastosowanie podejście bazujące na algorytmach metaheurystycznych w połączeniu ze sztucznymi sieciami neuronowymi. W publikacji tej został przeanalizowany problem optymalizacji przepływów finansowych podczas realizacji wielomodalnego przedsięwzięcia budowlanego. W przedstawianym modelu uwzględniono zarówno koszty (bezpośrednie i pośrednie) realizacji, wpływy pieniężne wynikające z umowy, jak i dodatkowe ograniczenia w postaci terminu dyrektywnego.

W publikacji (Podolski & Sroka, 2019) przedstawiono model optymalizacji harmonogramów przedsięwzięć wieloobiektowych. W funkcji celu uwzględniono koszty całkowite (koszty bezpośrednie, koszty pośrednie, koszty związane z niedotrzymaniem terminów dyrektywnych, koszty przestoju brygad roboczych). Dla danego uszeregowania zastosowano programowanie matematyczne do znalezienia rozwiązania optymalnego. Uwzględniono również możliwość optymalizacji kolejności wykonania obiektów z zastosowaniem metody symulowanego wyżarzania. Dokonano analizy eksperymentalnej proponowanego podejścia oraz zastosowano model na przykładzie rzeczywistym.

W publikacji (Bożejko i in., 2019) rozważany jest problem harmonogramowania robót budowlanych uwzględniający minimalizację sumy kar za przekroczenie terminu dyrektywnego wykonania poszczególnych obiektów budowlanych. Parametry w modelu są określone za pomocą liczb rozmytych lub zmiennych losowych. Do rozwiązania problemu został zastosowany algorytm przeszukiwania z zabronieniami. Skuteczność działania metody została przedstawiona na przykładzie realizacji kilkunastu apartamentowców wchodzących w skład nowopowstającego osiedla.

W pracy (Sroka i in., 2021) zaprezentowano optymalizacyjny model pozwalający optymalizować przepływy finansowe przy realizacji wieloobiektowych przedsięwzięć wielomodalnych. W modelu uwzględniono różnego rodzaju koszty (bezpośrednie, pośrednie, kary za np. niedotrzymanie terminów dyrektywnych, wynikające z przesuniętych okresów płatności) oraz ograniczenia technologiczno-organizacyjne. Do rozwiązania problemu zastosowano wieloetapowe algorytmu metaherustyczne bazujące na algorytmie symulowanego wyżarzania oraz na algorytmach genetycznych.

W publikacji (Kulejewski i in., 2021) zaproponowano rozszerzenie metody łańcucha krytycznego do planowania realizacji przedsięwzięć wieloobiektowych. Nowością zaproponowanego podejścia jest dwuetapowe wymiarowanie buforów czasowych z zastosowaniem modelu optymalizacyjnego deterministycznego oraz stochastycznego. Jako kryterium optymalizacji wybrano funkcję opartą na przepływach finansowych projektu budowlanego. Zaproponowana metoda została przetestowana na przykładzie realizacji farmy wiatrowej składającej się z trzech turbin. Przedstawiona w artykule metoda okazała się skuteczniejsza od tradycyjnej metody łańcucha krytycznego.

W pracy (Milat i in., 2021) zostało zaproponowane podejście, które jest złożoną kompromisową analizą pomiędzy czasem trwania przedsięwzięcia, jego zyskiem oraz odpornością systemu. W metodzie uwzględniono trzy funkcje celu: czas, zysk i odporność harmonogramu przedsięwzięcia budowlanego i sformułowano funkcję metakryterium. W wyniku optymalizacji powstaje proaktywny harmonogram bazowy. Zaproponowana metoda została przetestowana na przedsięwzięciu budowlanym składającym się z 8 prac budowlanych. Autorzy zwracają uwagę, że powinno się przetestować metodę na większych przedsięwzięciach.

W tabeli 3 przedstawiono najważniejsze publikacje uwzględniające analizę kosztową. Publikacje zostały uporządkowane w sposób chronologiczny. Pojęcie uwzględnienia ograniczeń w sposób elastyczny został wytłumaczony w rozdziale 3.2.

Tabela 3: Wybrane publikacje uwzględniające analizę kosztową w porządku chronologicznym

Autor, rok	Podejście/metoda	Zastosowanie	Ograniczenia metody
Moselhi i Elrayes, 1993	Deterministyczne, dwuetapowe programowanie dynamiczne	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny
Hegazy i Wassef, 2001	Deterministyczne, algorytmy genetyczne	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów bezpośrednich, pośrednich, nagród za wcześniejsze wykonanie, kar za opóźnienia, przestojów oraz likwidacji szkód

Marcinkowski, 2002	Deterministyczne, metoda podziału i ograniczeń	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów za niedotrzymanie terminów dyrektywnych oraz kosztów przerw w pracy brygad
Senouci i Eldin, 2004	Deterministyczne, rozszerzony algorytm genetyczny	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w funkcji celu jako funkcja kary. Podejście nieelastyczne. Uwzględnienie jedynie kosztów całkowitych
Mika, Waligóra i Węglarz, 2004	Deterministyczne, algorytm symulowanego wyżarzania oraz przeszukiwanie z zabronieniami	Harmonogramowanie przedsięwzięć wielomodalnych na przykładach teoretycznych	Uwzględnia jedynie przepływy pieniężne. Ograniczenia uwzględnione w sposób nieelastyczny. Brak uwzględnienia kosztów niedotrzymania terminów i kosztów przerw w pracy brygad
El-Rayes i Kandil, 2005	Deterministyczne, algorytmy genetyczne	Harmonogramowanie realizacji przedsięwzięć liniowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów całkowitych
Liu i Wang, 2008	Deterministyczne, dwuetapowa optymalizacja, programowanie ograniczeń	Harmonogramowanie realizacji mostu	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów bezpośrednich i pośrednich
Rogalska, Bożejko i Hejducki, 2008	Deterministyczne, hybrydowy algorytm ewolucyjny	Praca teoretyczna	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów bezpośrednich
San Cristobal, 2009	Deterministyczne, algorytm binarny oraz algorytm genetyczny	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów całkowitych
Ezeldin i Soliman, 2009	Deterministyczny, algorytm genetyczny i programowanie dynamiczne	Budowa zespołu stacji paliw	Brak uwzględnienia ograniczeń
Jaśkowski i Biruk, 2014	Deterministyczne, programowanie mieszane (liniowo-całkowitoliczbowe)	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów całkowitych, brak możliwości zmiany kolejności wykonania obiektów
Zhang, Zou i Qi, 2015	Deterministyczne, programowanie mieszane nieliniowe z zastosowaniem logiki miękkiej, algorytmy genetyczne	Harmonogramowanie realizacji mostu	Ograniczenia uwzględnione w sposób nieelastyczny
Podolski, 2016	Deterministyczny, algorytm symulowanego wyżarzania.	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnienie jedynie kosztów całkowitych

Duc-Hoc, Duc-Long, Minh-Tin, Trong-Nhan i Anh-Duc, 2017	Deterministyczny, Oposition Multiple Objective Symbiotic Organisms Search (OMOSOS)	Realizacja betonowego mostu, harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględnione jedynie niektóre koszty, ograniczenie jedynie w ciągłości pracy brygad
Rosłon i Kulejewski, 2019	Deterministyczne, algorytmy metaheurystyczne i sieci neuronowe	Realizacja obiektu budowlanego, przedsięwzięcie wielomodalne	Ograniczenia jedynie w postaci relacji sieciowych oraz terminu dyrektywnego
Podolski i Sroka, 2019	Deterministyczne, programowanie liniowe, symulowane wyżarzanie	Harmonogramowanie przedsięwzięć wieloobiektowych	Uwzględniono jedynie koszty, nie uwzględniono nagród za np. wcześniejsze wykonanie obiektów, ograniczenia jedynie w postaci relacji sieciowych uwzględnione w sposób nieelastyczny
Bożejko, Hejducki i Wodecki, 2019	Parametry są liczbami rozmytymi i liczba losowymi, przeszukiwanie z zabronieniami	Harmonogramowanie przedsięwzięć wieloobiektowych, kompleks kilkunastu obiektów	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględniono jedynie kary za niedotrzymanie terminów dyrektywnych
Sroka i inni, 2021	Deterministyczne, wieloetapowy algorytm metaheurystyczny	Harmonogramowanie przedsięwzięć wieloobiektowych	Ograniczenia uwzględnione w sposób nieelastyczny
Kulejewski, Ibadov, Rosłon i Zawistowski, 2021	Optymalizacja deterministyczna i stochastyczna	Harmonogramowanie przedsięwzięć wieloobiektowych, farma wiatrowa	Ograniczenia uwzględnione w sposób nieelastyczny, uwzględniono jedynie przepływy finansowe
Milat, Knezic i Sedlar, 2021	Deterministyczne, programowanie matematyczne oparte na metakryterium	Harmonogramowania robót budowlanych, przykład małego przedsięwzięcia budowlanego	Ograniczenia uwzględnione w sposób nieelastyczny, w kosztach uwzględniono jedynie koszty bezpośrednie i koszty niedotrzymania terminów dyrektywnych

2.3.5. Prace doktorskie i monografie

W celu uzupełnienia przeglądu literatury postanowiono wyodrębnić prace o charakterze dysertacji doktorskich oraz monografie w zakresie planowania przedsięwzięć wieloobiektowych lub realizujących koncepcję harmonogramowania interaktywnego.

W dysertacji (Podolski, 2008) autor wziął pod uwagę problem planowania robót budowlanych w systemie pracy potokowej. Zostały opracowane nowe modele pracy potokowej uwzględniające relację kolejnościową, wyrażoną w postaci sekwencji oraz w postaci grafu. Dodatkowo wykorzystano modele uwzględniające jedną oraz wiele grup roboczych wykonujących robotę jednego rodzaju. Analizowano również różne funkcję celu: czas realizacji przedsięwzięcia oraz dwukryterialną funkcję koszt/czas. W ograniczeniach wzięto pod uwagę

możliwość „zazębiania się” robót, występowania przerw technologicznych oraz dodatkowe czasy transportu brygad roboczych między obiektami. Weryfikacja modeli została przeprowadzona na przykładach budowlanych przedsięwzięć wieloobiektowych. W pracy nie zostały uwzględnione przede wszystkim: koszty niedotrzymania terminów dyrektywnych, koszty nieciągłości pracy brygad, ograniczenia w sposób elastyczny.

Rozprawa doktorska (Krawczyńska-Piechna, 2015) skupia się na procesach decyzyjnych w planowaniu przedsięwzięć budowlanych z wykorzystaniem konstrukcji tymczasowych, zwłaszcza w monolitycznym budownictwie betonowym. Celem badań jest opracowanie spójnej metody programowania przebiegu prac budowlanych z analizą wykorzystania konstrukcji tymczasowych, w celu osiągnięcia maksymalnej efektywności kosztowej. Zaproponowano technikę symulacji komputerowej z wykorzystaniem wiedzy eksperckiej i algorytmów szeregowania zadań z ograniczeniami. Na początku pracy identyfikowane są problemy i ograniczenia technologiczno-organizacyjne w monolitycznym budownictwie betonowym, a następnie analizowane są osiągnięcia naukowe w obszarze planowania robót betonowych, wraz z możliwościami wykorzystania innych podejść i technik. Praca proponuje algorytm interaktywnej symulacji komputerowej przebiegu robót, który pozwala na eksperymentowanie planistyczne i mierzenie efektywności wykorzystania konstrukcji pomocniczych przy budowie obiektu. Zaproponowana metoda planowania została zweryfikowana na rzeczywistej sytuacji planistycznej budownictwa. W pracy zaprezentowano podejście interaktywnego planowania, które było dla autora inspiracją do opracowanej metody priorytetowego harmonogramowania.

Monografia (Hejducki & Rogalska, 2017) przedstawia rozwiązania problemów z praktyki budowlanej harmonogramowania wieloobiektowych przedsięwzięć budowlanych, przy wykorzystaniu aplikacji MS Excel i Metody Sprzężeń Czasowych (TCM). Autorzy rozwiązali zagadnienia wynikające z metodyki Petersburskiej Szkoły Potokowej Organizacji Robót Budowlanych, kontynuując problematykę zapoczątkowaną przez Profesora Afanaseva. Praca zawiera nowe modele i algorytmy obliczeniowe oraz metodykę TCM do harmonogramowania przedsięwzięć budowlanych. Autorzy rozwijają Metody Sprzężeń Czasowych, a także prezentują metodę harmonogramowania robót budowlanych z uwzględnieniem czynników losowych metodą Goldratta. Opracowane modele i algorytmy obliczeniowe są przedstawione na licznych przykładach ilustrujących, jak mogą być wykorzystane w obliczeniach inżynierskich z wykorzystaniem aplikacji MS Excel.

Rozprawa doktorska (Tomczak, 2020) podjęła problem harmonizacji harmonogramu budowlanego przedsięwzięcia wieloobiektowego. W procesie optymalizacji redukuje się cykl budowy poszczególnych obiektów, czas realizacji całego przedsięwzięcia oraz przerwy w pracy brygad roboczych. W modelu uwzględniono również metodę grupowego wspomagania decyzji

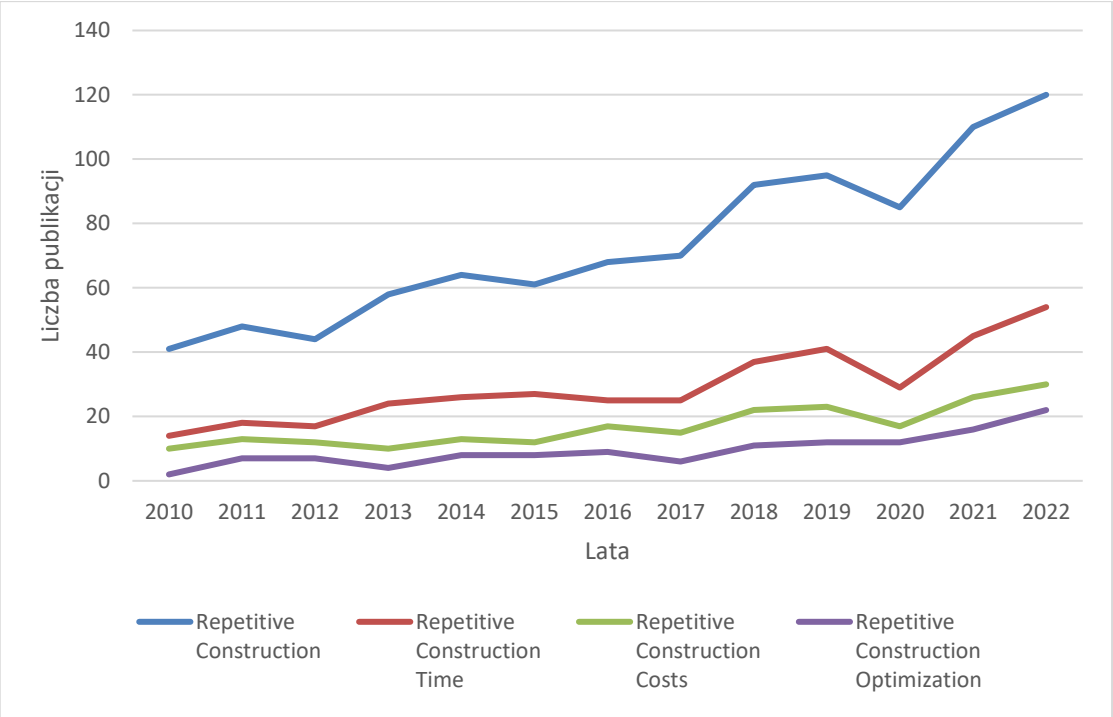
do wyboru istotności kryteriów oceny harmonogramów, wykorzystując zmodyfikowaną wersję metody AHP oraz metodę przeglądu wiązką światła (Light Beam Search). W celu znalezienia rozwiązania suboptymalnego został zastosowany algorytm przeszukiwania rojem cząstek. Algorytm zaimplementowano jako program komputerowy. Autor postuluje rozwinięcie modelu o wartości niedeterministyczne oraz rozbudowanie programu o interfejs graficzny użytkownika.

Dysertacja (Kostrzewa-Demczuk, 2022) rozszerza zastosowania klasycznej metody sprzężeń czasowych TCM. Autorka postanowiła uwzględnić podejście probabilistyczne zamiast deterministycznego. Dla wszystkich najbardziej popularnych metod TCM (TCM I, TCM II oraz TCM III) zostały opracowane metody nazwane PTCM, uwzględniające czas jako rozkład zmiennych losowych. Metoda pozwala wyznaczyć minimalny, najbardziej prawdopodobny i maksymalny czas realizacji wieloobiektowego przedsięwzięcia budowlanego. Dużą zaletą pracy jest zaimplementowanie proponowanych metod w popularnym programie Excel. Poszerza to zasięg i dostępność do opracowanej metody inżynierom. Skuteczność opracowanych metod została sprawdzona na rzeczywistej realizacji wieloobiektowej. Poza wskazanymi w dysertacji kierunkami dalszych badań (takich jak opracowanie nowej metody predykcji danych wejściowych lub opracowanie nowego i bardziej przyjaznego dla użytkownika interfejsu komputerowego), zasadnym wydaje się wskazanie rozszerzenia metody o PTCM IV, V, VI oraz uwzględnienie analizy i wpływu kosztów na realizację wieloobiektowego przedsięwzięcia budowlanego.

2.3.6. Analiza liczby publikacji w bazie Scopus

Dodatkowo postanowiono przeprowadzić analizę liczby artykułów z zakresu harmonogramowania powtarzalnych przedsięwzięć budowlanych przy wykorzystaniu wyszukiwarki Scopus. Wzięto pod uwagę tylko pola „tytuł", „abstrakt" i „słowa kluczowe" oraz uwzględnić tylko publikacje z grupy „inżynieria". Pod hasłem „repetetive" & „construction" występuje 1556 publikacji. Po zawężeniu wyników do takich, które zawierają słowo „costs" liczba publikacji wynosi 352, po zawężeniu używając słowa „time" liczba publikacji wynosi 565, natomiast używając słowo „optimization" otrzymano 176 wyników[6]. Dodatkowo na rysunku 9 przedstawiono liczbę publikacji w poszczególnych latach, uwzględniając opisane powyżej opcje wyszukiwania. Można zauważyć, że zagadnienie planowania i realizacji przedsięwzięć powtarzalnych jest często poruszane przez naukowców na całym świecie. Dotyczy to zarówno analizy czasowej, kosztowej jak i zagadnień optymalizacyjnych. Ponadto można zauważyć tendencję rosnącą, jeżeli chodzi o zainteresowanie naukowców wspomnianym zagadnieniem.

[6] Data dostępu 19.03.2023 r.



Rysunek 9: Liczba publikacji w zależności od wyszukiwanych słów kluczowych w przeglądarce Scopus w obszarze badań inżynierskich (data dostępu 19.03.2023 r.)

2.4. Podsumowanie uzasadnienia wyboru tematu

Najważniejsze wnioski sformułowane na podstawie wywiadów przeprowadzonych z inżynierami, kierownikami oraz osobami decyzyjnymi w przedsiębiorstwach budowlanych, przeglądu literatury, analizy liczby publikacji w bazie Scopus, analiz rynku deweloperskiego to:

- brak świadomości inżynierów, że kolejność wykonania obiektów ma wpływ na czas i koszt przedsięwzięcia wieloobiektowego;
- braki w oprogramowaniu pomocnym przy harmonogramowaniu przedsięwzięć wieloobiektowych;
- rosnący trend liczby wykonywanych przedsięwzięć wieloobiektowych, zarówno jako obiektów mieszkaniowych, jak i biurowych realizowanych przez deweloperów, ale też podmioty indywidualne;
- brak modeli uwzględniających wszystkie rodzaje kosztów, ważnych z punktu widzenia realizacji przedsięwzięcia wieloobiektowego;
- brak modeli, które w sposób elastyczny pozwolą definiować ograniczenia technologiczno-organizacyjne;
- zainteresowanie naukowców harmonogramowaniem przedsięwzięć wieloobiektowych co uzasadnia, że temat jest ważny i aktualny.

W przeglądzie literatury udowodniono brak publikacji uwzględniających kompleksowe podejście do modelowania kosztów realizacji przedsięwzięć wieloobiektowych przy ograniczeniach technologiczno-organizacyjnych. Żaden z modeli nie uwzględnia przy harmonogramowaniu wieloobiektowych przedsięwzięć budowlanych jednocześnie: kosztów bezpośrednich, kosztów pośrednich, kar za niedotrzymanie terminów dyrektywnych, kar za niedotrzymanie ciągłości pracy brygad, premii za wcześniejsze wykonanie obiektów, ograniczeń w dostępie brygad i obiektów, ograniczeń technologicznych i organizacyjnych. Brak jest również prac, w których ograniczenia technologiczno-organizacyjne uwzględnione są w sposób elastyczny. Tę opinię potwierdza prof. Marcinkowski. W publikacji (Marcinkowski, 2017) twierdzi, że:

*„(…) potrzebna jest metoda planowania tego rodzaju przedsięwzięć (*modelowania ograniczeń w harmonogramowaniu pracy brygad w systemie pracy potokowej – przyp. aut.*), elastyczna pod względem definiowania ograniczeń i kryteriów optymalizacji harmonogramów. (…) Elastyczna powinna być też funkcja celu. Nie jest bowiem prawdą, że w planowaniu najczęściej chodzi nam o jak najszybsze zrealizowanie przedsięwzięcia.”*

Powyższy cytat wskazuje kierunek rozwoju harmonogramowania przedsięwzięć wieloobiektowych.

Wstępne badania pilotażowe wykazały, że nie istnieje skuteczne narzędzie informatyczne wspomagające proces harmonogramowania przedsięwzięć wieloobiektowych, modelujące w sposób elastyczny zależności technologiczno-organizacyjne występujące podczas realizacji tego typu przedsięwzięć.

Przeprowadzone analizy udowodniły także, iż rośnie liczba realizacji wieloobiektowych (na przykładzie przedsiębiorstw ATAL S.A., Dom Development, Echo Investment oraz danych GUS). Można zauważyć dużą liczbę realizacji wieloobiektowych na przykładzie Krakowa (osiedle przy ul. Sosnowieckiej, Szpital Uniwersytecki w Prokocimiu, Osiedle Avia, obiekty biurowe High-Five, O3 Business Campus przy ul. Opolskiej, i inne) co potwierdza aktualnie panujące trendy w budownictwie.

Podsumowując, podjęty temat rozprawy jest niezwykle istotny i obecnie brakuje narzędzi do jego rozwiązania. Wpisuje się on także w aktualne trendy naukowe w dziedzinie harmonogramowania przedsięwzięć wieloobiektowych.

2.5. Teza badawcza

W rozprawie doktorskiej sformułowano następującą tezę:

> Opracowana metoda priorytetowego harmonogramowania, uwzględniająca elastyczne ograniczenia technologiczno-organizacyjne oraz możliwość zmiany kolejności wykonania obiektów, pozwala na minimalizację kosztów całkowitych wieloobiektowych przedsięwzięć budowlanych.

2.6. Cele rozprawy doktorskiej

W rozprawie doktorskiej postawiono następujące cele:

- opracowanie czasowo-kosztowego modelu harmonogramowania priorytetowego, minimalizującego koszty całkowite oraz uwzględniającego ograniczenia organizacyjno-technologiczne w sposób elastyczny, służącego do harmonogramowania wieloobiektowych przedsięwzięć budowlanych;
- opracowanie metody optymalizacji dyskretnej, wspomagającej harmonogramowanie wieloobiektowych przedsięwzięć budowlanych z uwzględnieniem kolejności realizacji obiektów;
- implementacja komputerowa opracowanego modelu, pozwalająca na zautomatyzowanie obliczeń.

3. Metoda priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych

3.1. Specyfika harmonogramowania wieloobiektowych przedsięwzięć budowlanych

Na rysunku 10 przedstawiono uproszczony podział przedsięwzięć budowlanych ze względu na metodę organizacji pracy.

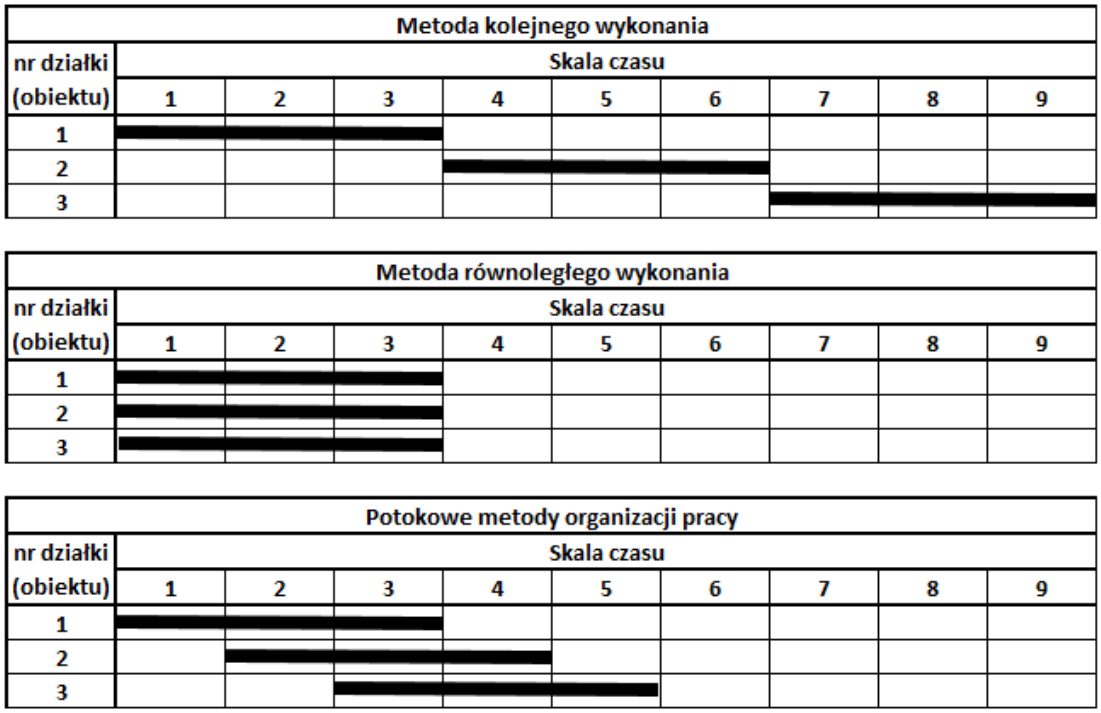
Rysunek 10: Podział przedsięwzięć budowlanych ze względu na metody organizacji pracy. Opracowanie własne na podstawie (Jaworski, 2009)

Każde przedsięwzięcie budowlane można zrealizować metodą kompleks operacji. Jako kompleks operacji należy rozumieć: „taki zbiór operacji, z których każda ma swój początek, koniec, czas realizacji oraz jest umiejscowiona w strukturze sieciowej (Józewczyk, 2001)" lub bardziej ogólnie jako system składający się ze zbioru zasobów, zbioru operacji oraz kryterium optymalności (Węglarz, 1981). Spośród wszystkich przedsięwzięć można wyróżnić te, które mogą być realizowane:

- metodą kolejnego wykonania;
- metodą równoległego wykonania;

- potokowymi metodami organizacji pracy.

Na rysunku 11 przedstawiono symbolicznie wspomniane metody organizacji.



Rysunek 11: Schematy metod realizacji przedsięwzięć budowlanych

Metoda kolejnego wykonania polega na realizacji procesów roboczych jeden po drugim. Jest to wyjątkowo korzystne w przypadku posiadania niewielkich zasobów do realizacji przedsięwzięcia, jednak wydłuża to zdecydowanie czas jego realizacji.

Metoda równoległego wykonania polega na realizacji wszystkich procesów w jednym czasie. Zaletą tej metody jest krótki czas realizacji przedsięwzięcia, wadą zaś konieczność posiadania dużych zasobów. Ponadto, nie zawsze istnieje możliwość wykorzystania tej metody (np. nie można robić jednocześnie wszystkich kondygnacji budynku).

Potokowa metoda organizacji pracy (zwana często systemem pracy potokowej lub metodą pracy równomiernej) została zaczerpnięta z przemysłu. W metodzie tej przedsięwzięcie jest podzielone na działki robocze, tworzące zbiór:

$$O = \{1,2, \dots, i, \dots n\},$$

gdzie n będzie oznaczało liczbę działek roboczych.

W niniejszej rozprawie działki robocze będą utożsamiane z obiektami budowlanymi. Jeżeli nie wspomniano inaczej, to każdy obiekt budowlany będzie oddzielną działką roboczą.

Dodatkowo na każdej działce (obiekcie) przewidziany jest do wykonania zbiór procesów roboczych:

$$P_i = \{p_{i,1}, p_{i,2}, \dots, p_{i,j}, \dots, p_{i,m}\}, i = \{1,2, \dots, n\},$$

gdzie m będzie oznaczało liczbę wyspecjalizowanych brygad. Procesy będą też numerowane jako $OiBj$ gdzie i oznacza numer obiektu, a j numer brygady, która realizuje dany proces.

Zbiór procesów roboczych realizowany jest w porządku technologicznym, czyli:

$$p_{i,j} \prec p_{i,j+1}, j = \{1,2, \dots, m-1\}$$

Każdy proces roboczy jest realizowany przez wyspecjalizowaną brygadę roboczą.

W niniejszej rozprawie poprzez wyspecjalizowaną brygadę należy rozumieć jednostkę organizacyjną realizującą proces roboczy, przemieszczającą się po obiektach budowlanych w ustalonej kolejności. Brygada może być utożsamiana z przedsiębiorstwem podwykonawczym wykonującym dane procesy robocze (Marcinkowski, 2002). Poprzez proces roboczy należy rozumieć prosty proces budowlany. Dla uproszczenia, proces roboczy będzie nazywany po prostu procesem, a wyspecjalizowana brygada, po prostu brygadą.

Na rysunku 12 przedstawiono poglądowo ideę potokowej metody pracy. Brygada B1 wykonuje proces $O1B1$ na obiekcie O1. Po jej zakończeniu brygada B1 przenosi się na obiekt O2, natomiast na jej miejsce wchodzi brygada B2 i wykonuje proces $O1B2$. Po zakończeniu robót przez brygady B1 i B2 przenoszą się one na kolejny obiekt, natomiast na obiekt O1 wchodzi brygada B3 i wykonuje proces $O1B3$. Schemat powtarza się do momentu, kiedy brygada B1 skończy realizować proces na ostatnim obiekcie O3 i opuści plac budowy. Prace trwają do zakończenia procesu przez brygadę B4 na obiekcie O3.

Rysunek 12: Poglądowe przedstawienie idei potokowych metod organizacji

Potokowe metody organizacji pracy można zastosować przy realizacji jednego obiektu budowlanego, jednak w pracy podjęto problem harmonogramowania przedsięwzięć wieloobiektowych (co najmniej 2). Przedsięwzięcia wieloobiektowe realizowane potokowymi metodami organizacji pracy można podzielić na trzy rodzaje:

- **Obiekty jednego typu**: obiekty o jednakowej wielkości, na których wykonuje się analogiczne konstrukcje oraz pozostałe zadania budowlane, stosując jednakowe

technologie. Przedmiary robót dla wszystkich obiektów są identyczne. Jako przykład przedsięwzięcia wieloobiektowego tego typu można podać budowę osiedla domów jedno- lub wielorodzinnych. Oczywiście przyjęcie jednakowych przedmiarów we wszystkich obiektach jest uproszczeniem, ponieważ poszczególne obiekty mogą różnić się ze względu na przystosowanie ich do wymagań przyszłych mieszkańców. Wpływ tych zmian jest jednak pomijalny;

- **Obiekty jednorodne**: obiekty realizowane za pomocą jednakowej technologii, które różnią się długością, powierzchnią lub przestrzenią zabudowy. Dzięki jednorodności obiektów występuje proporcjonalność pomiędzy wielkościami i pracochłonnością robót. Przykładem może być realizacja obiektów liniowych. Całość realizacji może dzielić się na działki robocze o różnych wielkościach (długościach);
- **Obiekty niejednorodne**: obiekty realizowane jednakową technologią, w których potrzeby użytkowe zdecydowały o zmianach wielkości rzutu, o różnych rozpiętościach, wysokościach, przekrojach elementów konstrukcyjnych. Przedmiarów nie charakteryzuje proporcjonalność. Jest to typ najbardziej ogólny. Za przykład może posłużyć realizacja kompleksu obiektów mieszkalnych wielorodzinnych, w którego skład wchodzą zarówno bloki niskie, jak i wysokie (Jaworski, 2009)(Rowiński, 1982).

W niniejszej rozprawie będą brane pod uwagę tylko obiekty niejednorodne. Są one najbardziej ogólnym rodzajem obiektów realizowanych w ramach przedsięwzięć wieloobiektowych.

Stosowanie potokowych metod organizacji pracy przynosi korzyści, które przemawiają za ich wykorzystaniem:

- każda brygada wykonuje ten sam proces na każdym z obiektów. Dzięki temu nie jest ona zmuszana do ciągłej zmiany charakteru wykonywanej pracy i osiąga większą wydajność. Jeżeli brygadą będzie przedsiębiorstwo podwykonawcze, to będzie wykonywała ona tylko specjalistyczne prace, w zakresie których posiada duże doświadczenie;
- przy odpowiedniej organizacji pracy można wyeliminować przerwy w realizacji procesów poszczególnych brygad i/lub przerwy w realizacji procesów na poszczególnych obiektach;
- można w prosty sposób zorganizować pracę, gdy konieczne jest zrealizowanie przedsięwzięcia budowlanego w krótszym czasie. Zatrudniając dodatkową brygadę istnieje możliwość wyznaczenia jej realizacji procesów na innym obiekcie lub (dla brygad o tej samej specjalizacji) realizacji procesów równolegle na jednym

obiekcie wraz z inną brygadą, ale w różnych jego częściach (dokonać podziału obiektu na działki robocze);

- realizacje wykonywane systemem pracy potokowej charakteryzują się korzystnym zaangażowaniem zasobów (finansowych, ludzkich oraz sprzętowych). Na początku realizacji zużycie zasobów jest stosunkowo małe ze względu na to, że procesy są wykonywane na niewielkiej liczbie obiektów. Z czasem fronty robót zostają otwarte na wielu obiektach i zaangażowanie zasobów osiąga stały, względnie niezmienny poziom. W miarę zbliżania się do zakończenia przedsięwzięcia zużycie zasobów maleje, ponieważ kończy się realizacja procesów na kolejnych obiektach, aż do zrealizowania wszystkich obiektów.

3.2. Koncepcja harmonogramowania priorytetowego

Podstawą priorytetowego harmonogramowania przedsięwzięć wieloobiektowych jest redefiniowany sposób określenia sprzężeń czasowych. W klasycznej metodzie TCM sprzężenia czasowe określają odległość między odpowiednimi charakterystykami procesów. Może to skutkować powstaniem redundancji ograniczeń i brakiem możliwości dochowania wszystkich narzuconych ograniczeń technologiczno-organizacyjnych, w wyniku czego niemożliwym jest znalezienie rozwiązania dopuszczalnego przy nadmiernych ograniczeniach. W przeciwieństwie do klasycznej metody TCM (Hejducki, 2004; Hejducki & Rogalska, 2011; Mrozowicz, 1997) zaproponowano podejście elastyczne.

Poprzez podejście elastyczne należy rozumieć takie podejście do harmonogramowania, które nie narzuca ograniczeń technologiczno-organizacyjnych w sposób sztywny. Oznacza to, że część narzuconych ograniczeń może nie zostać dochowana i jest to cecha tego podejścia. W celu realizacji podejścia elastycznego zaproponowano koncepcję priorytetowego harmonogramowania, która pozwala rozwiązać problem przed jakimi stoi decydent próbujący dotrzymać wszystkich ograniczeń narzuconych na planowane przedsięwzięcie. Wspomniany problem decyzyjny określenia ograniczeń technologiczno-organizacyjnych zostanie przedstawiony na poniższym przykładzie.

Zakłada się, że jest do zrealizowania przedsięwzięcie, podczas którego należy wykonać 4 różne procesy przez 4 brygady na 4 obiektach (budynkach). Czasy (bez jednostki) wykonania poszczególnych prac przez brygady przedstawiono w tabeli 4.

Tabela 4 Czasy wykonania poszczególnych procesów przez brygady - przykład ilustrujący metodę harmonogramowania priorytetowego

	B1	**B2**	**B3**	**B4**
O1	3	5	4	2
O2	2	4	3	4
O3	3	2	4	3
O4	2	4	5	1

Oczywistym jest, że zapewnienie ciągłości pracy brygad oraz (jako inny przykład) ciągłości pracy na obiektach nie nastręcza trudności. Są to znane z teorii sprzężeń czasowych ograniczenia. Terminy rozpoczęcia i zakończenia poszczególnych procesów dla ciągłości pracy brygad oraz ciągłości pracy na obiektach przedstawiono w tabelach 5 oraz 6.

Tabela 5 Ciągłość pracy brygad - przykład

O1B1		**O1B2**		**O1B3**		**O1B4**	
0	3	3	8	8	12	15	17
O2B1		**O2B2**		**O2B3**		**O2B4**	
3	5	8	12	12	15	17	21
O3B1		**O3B2**		**O3B3**		**O3B4**	
5	8	12	14	15	19	21	24
O4B1		**O4B2**		**O4B3**		**O4B4**	
8	10	14	18	19	24	24	25

Tabela 6 Ciągłość pracy na obiektach - przykład

O1B1		**O1B2**		**O1B3**		**O1B4**	
0	3	3	8	8	12	12	14
O2B1		**O2B2**		**O2B3**		**O2B4**	
6	8	8	12	12	15	15	19
O3B1		**O3B2**		**O3B3**		**O3B4**	
10	13	13	15	15	19	19	22
O4B1		**O4B2**		**O4B3**		**O4B4**	
13	15	15	19	19	24	24	25

Problem pojawia się jednak w przypadku narzucenia nadmiernej liczby ograniczeń. W przypadku narzucenia ograniczenia na roboty wykonywane przez brygadę 2, brygadę 3 oraz roboty wykonywane na obiekcie 3, niemożliwym staje się znalezienie rozwiązania tego zagadnienia. Można powiedzieć, że w tym momencie istnieje redundancja ograniczeń. Bez zmiany czasów trwania poszczególnych zadań, dotrzymanie ograniczeń jest niemożliwe.

Dotrzymując ciągłości pracy brygady 2 oraz ciągłości pracy na obiekcie 3 niemożliwym staje się dochowanie ciągłości pracy brygady 3, co zaprezentowano w tabeli 7 (termin rozpoczęcia roboty $O2B3$ musiałby być mniejszy niż termin zakończenia roboty $O2B2$ co jest niezgodne z założeniami).

Tabela 7 Ciągłość pracy dla brygady 2, brygady 3 i na obiekcie 3 - przykład

O1B1		**O1B2**		**O1B3**		**O1B4**	
0	3	3	8				
O2B1		**O2B2**		**O2B3**		**O2B4**	
3	5	8	12	11	14		
O3B1		**O3B2**		**O3B3**		**O3B4**	
9	12	12	14	14	18	18	21
O4B1		**O4B2**		**O4B3**		**O4B4**	
12	14	14	18				

Z tego powodu została opracowana koncepcja priorytetowego harmonogramowania. Koncepcja ta zakłada, że ograniczenia technologiczno-organizacyjne są określone w sposób elastyczny oraz powinny być dochowane „od najważniejszego do najmniej ważnego". Głównym założeniem koncepcji jest to, że nie wszystkie ograniczenia muszą (ale mogą, jeżeli jest to możliwe) być dotrzymane. Koncepcja ta nie zakłada usuwania żadnego narzuconego ograniczenia, ale to decydent określa ranking ograniczeń w postaci listy priorytetów (stąd nazwa metody).

Na przykład decydent mógłby ustalić następujący priorytet ograniczeń:

1) ciągłość pracy brygady 2;
2) ciągłość pracy brygady 3;
3) ciągłość pracy na obiekcie 3.

Wtedy rozwiązanie mogłoby wyglądać tak jak zaprezentowane w tabeli 8. Nie udało się niestety dotrzymać ciągłości pracy na obiekcie 3, jednak miała ona najmniejszy priorytet. Ciągłość pracy brygady 2 i brygady 3 została dotrzymana.

Tabela 8 Rozwiązanie z uwzględnieniem priorytetów - przykład

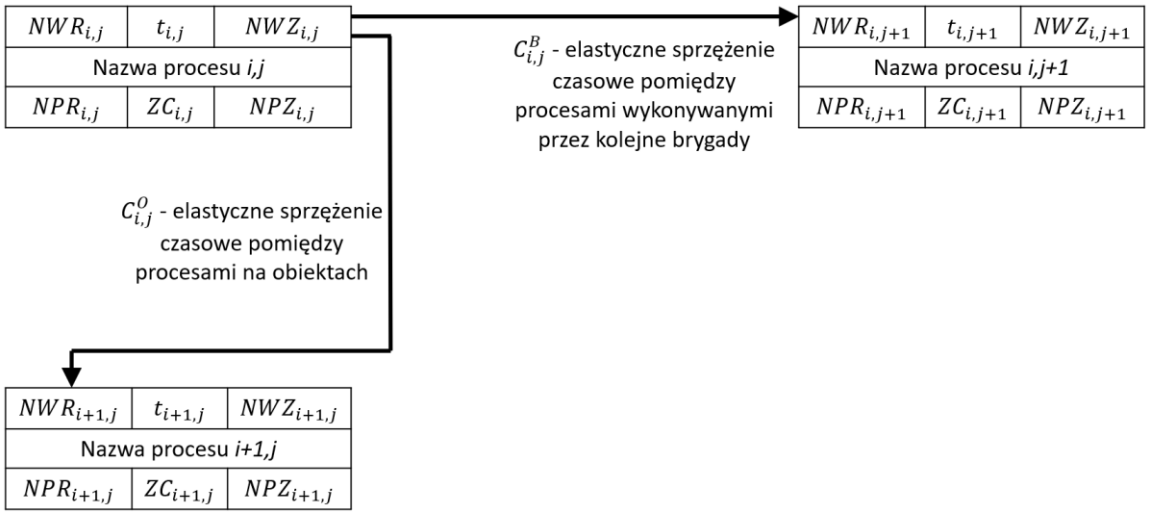
O1B1		**O1B2**		**O1B3**		**O1B4**	
0	3	3	8	8	12	12	14
O2B1		**O2B2**		**O2B3**		**O2B4**	
3	5	8	12	12	15	15	19
O3B1		**O3B2**		**O3B3**		**O3B4**	
9	12	12	14	15	18	19	23
O4B1		**O4B2**		**O4B3**		**O4B4**	
12	14	14	18	18	23	23	24

Podstawę koncepcji priorytetowego harmonogramowania stanowi Model Sieciowy Przedsięwzięcia (MSP) w wersji jednopunktowej (węzłowej). Węzeł MSP został przedstawiony na rysunku 13.

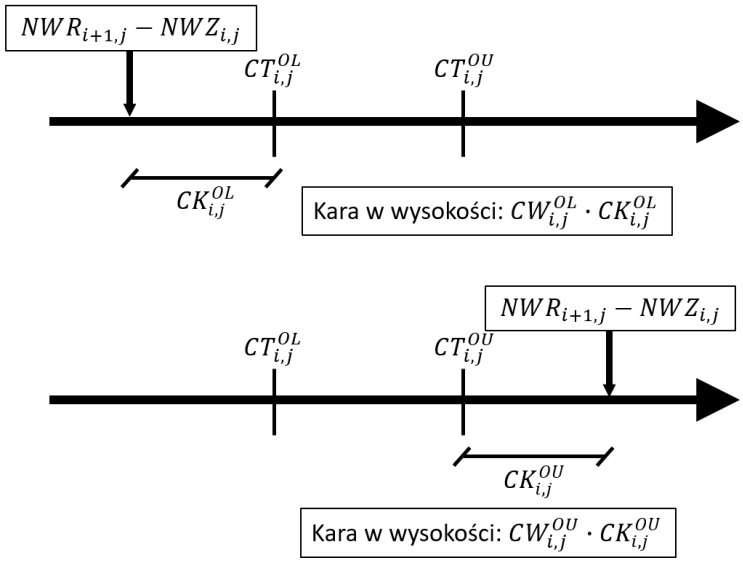
$NWR_{i,j}$	$t_{i,j}$	$NWZ_{i,j}$
Nazwa procesu *i,j*		
$NPR_{i,j}$	$ZC_{i,j}$	$NPZ_{i,j}$

Rysunek 13: Schemat procesu w MSP

Proces jest reprezentowany przez klasyczny węzeł w metodzie Critical Path Method (CPM). Posiada charakterystyki takie jak NWR (termin najwcześniejszego rozpoczęcia), NWZ (termin najwcześniejszego zakończenia), NPR (termin najpóźniejszego rozpoczęcia), NPZ (termin najpóźniejszego zakończenia) oraz ZC (całkowity zapas czasu). Każdy proces jest jednoznacznie określony przez parę (*i,j*). Indeks *i* oznacza numer obiektu, na którym dany proces jest realizowany, natomiast indeks *j* oznacza numer brygady, która będzie wykonywała dany proces. Węzły są połączone strzałkami symbolizującymi ustanowione sprzężenia elastyczne (w przeciwieństwie do metody CPM, w której strzałki symbolizują relacje). Dwa typy sprzężeń zostały przedstawione na rysunku 14.



Rysunek 14: Elastyczne sprzężenia czasowe $C^O_{i,j}$ oraz $C^B_{i,j}$.



Rysunek 15: Sposób naliczania kar dla sprzężeń $C^O_{i,j}$.

Elastyczne sprzężenie czasowe $C^O_{i,j}$ to sprzężenie pomiędzy najwcześniejszym terminem zakończenia procesu *i,j* oraz najwcześniejszym terminem rozpoczęcia procesu *i+1,j*. Jest to sprzężenie pomiędzy realizacją procesów tej samej brygady na kolejnych obiektach. Elastyczne sprzężenie czasowe $C^B_{i,j}$ to sprzężenie pomiędzy najwcześniejszym terminem zakończenia procesu *i,j* oraz najwcześniejszym terminem rozpoczęciem procesu *i,j+1*. Jest to sprzężenie pomiędzy realizacją procesów różnych brygad na tym samym obiekcie. Każde elastyczne sprzężenie czasowe $C^O_{i,j}$ będzie utożsamiane z parametrami jakie posiada:

$$C^O_{i,j} = (CT^{OL}_{i,j}, CW^{OL}_{i,j}, CT^{OU}_{i,j}, CW^{OU}_{i,j}),$$

oraz analogiczne dla sprzężeń $C^B_{i,j}$:

$$C^B_{i,j} = (CT^{BL}_{i,j}, CW^{BL}_{i,j}, CT^{BU}_{i,j}, CW^{BU}_{i,j}).$$

Poszczególne parametry oznaczają:

- $CT_{i,j}^{OL}$ - wartość **dolna** elastycznego sprzężenia czasowego pomiędzy procesami na kolejnych obiektach realizowanych przez tę samą brygadę;
- $CW_{i,j}^{OL}$ – jednostkowa kara za niedotrzymanie wartości dolnej sprzężenia $CT_{i,j}^{OL}$;
- $CT_{i,j}^{OU}$ – wartość **górna** elastycznego sprzężenia czasowego pomiędzy procesami na kolejnych obiektach realizowanych przez tę samą brygadę;
- $CW_{i,j}^{OU}$ – jednostkowa kara za niedotrzymanie wartości górnej sprzężenia $CT_{i,j}^{OU}$;
- $CT_{i,j}^{BL}$ - wartość **dolna** elastycznego sprzężenia czasowego pomiędzy procesami realizowanymi przez kolejne brygady na jednym obiekcie;
- $CW_{i,j}^{BL}$ – jednostkowa kara za niedotrzymanie wartości dolnej sprzężenia $CT_{i,j}^{BL}$;
- $CT_{i,j}^{BU}$ - wartość **górna** elastycznego sprzężenia czasowego pomiędzy procesami realizowanymi przez kolejne brygady na jednym obiekcie;
- $CW_{i,j}^{BU}$ – jednostkowa kara za niedotrzymanie wartości górnej sprzężenia $CT_{i,j}^{BU}$.

Parametr $CT_{i,j}^{OL}$ określa minimalną wartość, o jaką mogą różnić się charakterystyki $NWZ_{i,j}$ oraz $NWR_{i+1,j}$. A więc powinna zachodzić relacja:

$$NWR_{i+1,j} - NWZ_{i,j} \geq CT_{i,j}^{OL}.$$

Jeżeli jednak ta relacja nie zachodzi, a więc jeżeli:

$$NWR_{i+1,j} - NWZ_{i,j} < CT_{i,j}^{OL},$$

to dolne sprzężenie jest niedotrzymane. Niedotrzymanie wartości dolnej sprzężenia skutkuje naliczeniem kar w wysokości iloczynu parametru $CW_{i,j}^{OL}$ oraz stopnia niedotrzymania sprzężenia określanego jako $CK_{i,j}^{OL}$. Zostało to graficznie przedstawione na górnym schemacie na rysunku 15.

Parametr $CT_{i,j}^{OU}$ określa maksymalną wartość, o jaką mogą różnić się charakterystyki $NWZ_{i,j}$ oraz $NWR_{i+1,j}$. Analogicznie do powyższego przykładu powinna zachodzić relacja:

$$NWR_{i+1,j} - NWZ_{i,j} \leq CT_{i,j}^{OU}.$$

Jeżeli powyższa relacja nie zachodzi, a więc jeżeli:

$$NWR_{i+1,j} - NWZ_{i,j} > CT_{i,j}^{OU},$$

to górne sprzężenie jest niedotrzymane. Niedotrzymanie wartości górnej sprzężenia skutkuje naliczeniem kar w wysokości iloczynu parametru $CW_{i,j}^{OU}$ oraz stopnia niedotrzymania sprzężenia określanego jako $CK_{i,j}^{OU}$. Zostało to graficznie przedstawione na dolnym schemacie na rysunku 15.

Analogiczne stwierdzenia tyczą się elastycznych sprzężeń czasowych $C_{i,j}^{B}$. Parametr $CT_{i,j}^{BL}$ określa minimalną wartość, o jaką mogą różnić się charakterystyki $NWZ_{i,j}$ oraz $NWR_{i,j+1}$. Powinna więc zachodzić relacja:

$$NWR_{i,j+1} - NWZ_{i,j} \geq CT_{i,j}^{BL}.$$

Jeżeli jednak ta relacja nie zachodzi, a więc jeżeli:

$$NWR_{i,j+1} - NWZ_{i,j} < CT_{i,j}^{BL},$$

to dolne sprzężenie jest niedotrzymane. Niedotrzymanie wartości dolnej sprzężenia skutkuje naliczeniem kar w wysokości iloczynu parametru $CW_{i,j}^{BL}$ oraz stopnia niedotrzymania sprzężenia określanego jako $CK_{i,j}^{BL}$.

Parametr $CT_{i,j}^{BU}$ określa maksymalną wartość, o jaką mogą różnić się charakterystyki $NWZ_{i,j}$ oraz $NWR_{i,j+1}$. Powinna zachodzić relacja:

$$NWR_{i,j+1} - NWZ_{i,j} \leq CT_{i,j}^{BU}.$$

Jeżeli powyższa relacja nie zachodzi, a więc jeżeli:

$$NWR_{i,j+1} - NWZ_{i,j} > CT_{i,j}^{BU},$$

to górne sprzężenie jest niedotrzymane. Niedotrzymanie wartości górnej sprzężenia skutkuje naliczeniem kar w wysokości iloczynu parametru $CW_{i,j}^{BU}$ oraz stopnia niedotrzymania sprzężenia określanego jako $CK_{i,j}^{BU}$.

Pomiędzy parametrami musi zachodzić zależność:

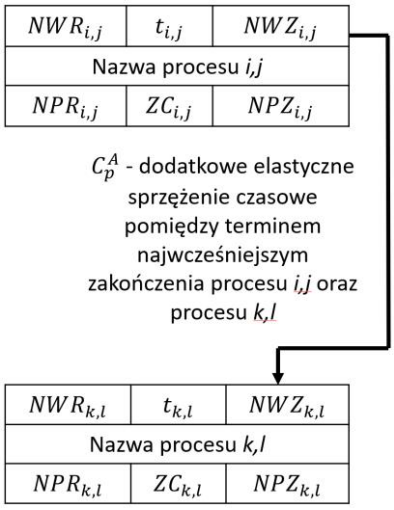
$$CT_{i,j}^{OL} \leq CT_{i,j}^{OU} \text{ oraz } CT_{i,j}^{BL} \leq CT_{i,j}^{BU}.$$

W tabeli 9 przedstawiono przykład obliczeniowy ilustrujący podane właściwości elastycznych sprzężeń czasowych $C_{i,j}^{O}$. W tabeli można zauważyć, jak kształtują się wartości niedotrzymania dolnego i górnego sprzężenia oraz kary naliczanej w funkcji celu w przypadku niedotrzymania sprzężenia dla różnych wartości charakterystyk procesów oraz różnej wartości parametrów.

Tabela 9: Przykład, który prezentuje właściwości sprzężeń czasowych elastycznych

Dane		Parametry				Obliczenia			
$NWZ_{i,j}$	$NWZ_{i+1,j}$	$CT_{i,j}^{OL}$	$CW_{i,j}^{OL}$	$CT_{i,j}^{OU}$	$CW_{i,j}^{OU}$	Niedotrzymanie wartości dolnej sprzężenia	Kara niedotrzymania dolnego sprzężenia	Kara wartości górnej sprzężenia	Koszt niedotrzymania górnego sprzężenia
5	5	0	200	0	100	0	0	0	0
5	4	0	200	0	100	1	200	0	0
5	6	0	200	0	100	0	0	1	100
5	5	4	200	10	100	4	800	0	0
5	5	0	200	4	100	0	0	4	400
5	4	-1	200	0	100	0	0	0	0
5	3	-1	200	0	100	1	200	0	0
5	4	-3	200	-2	100	0	0	1	100
5	10	0	200	100	100	0	0	0	0

Poza zdefiniowanymi elastycznymi sprzężeniami czasowymi należy też zdefiniować dodatkowe elastyczne sprzężenia czasowe. Jest to uogólnienie elastycznych sprzężeń czasowych. Dodatkowe elastyczne sprzężenia mogą dotyczyć różnych charakterystyk danego procesu, a nie tylko, jak wcześniej wspomniano, relacji pomiędzy terminem zakończenia następnika i terminem rozpoczęcia poprzednika. W publikacji (Mrozowicz, 1997) przedstawiono wiele różnych typów sprzężeń czasowych. Każdy z nich można przekształcić w sprzężenia elastyczne dodatkowe. Przykładowe sprzężenie przedstawiono na rysunku 16.



Rysunek 16: Przykładowe dodatkowe elastyczne sprzężenie czasowe

Dodatkowe elastyczne sprzężenie czasowe będzie utożsamiane z zestawem parametrów, jakimi się charakteryzuje:

$$C_p^A = (CT_p^{AL}, CW_p^{AL}, CT_p^{AU}, CW_p^{AU}, S_p, P_p), \text{ gdzie } p \in \{1,2 \dots u\}$$

gdzie:

- CT_p^{AL} – wartość **dolnego** dodatkowego elastycznego sprzężenia czasowego pomiędzy odpowiednimi charakterystykami procesów S_p oraz P_p;
- CW_p^{AL} – jednostkowa kara za niedotrzymanie dolnego sprzężenia CT_p^{AL};
- CT_p^{AU} - wartość **górnego** dodatkowego elastycznego sprzężenia czasowego pomiędzy odpowiednimi charakterystykami procesów S_p oraz P_p;
- CW_p^{AU} – jednostkowa kara za niedotrzymanie górnego sprzężenia CT_p^{AU};
- S_p – odpowiednia charakterystyka następnika dodatkowego sprzężenia czasowego procesu. $S_p \in \{NWR_{i,j}; NWZ_{i,j}; NPR_{i,j}; NPZ_{i,j}; ZC_{i,j}\}$;
- P_p – odpowiednia charakterystyka następnika dodatkowego sprzężenia czasowego procesu. $P_p \in \{NWR_{k,l}; NWZ_{k,l}; NPR_{k,l}; NPZ_{k,l}; ZC_{k,l}\}$.

Elastyczne sprzężenia czasowe C_p^A działają analogicznie do elastycznych sprzężeń czasowych $C_{i,j}^O$ oraz $C_{i,j}^B$. Parametr CT_p^{AL} określa minimalną wartość, o jaką mogą różnić się charakterystyki S_p oraz P_p:

$$S_p - P_p \geq CT_p^{AL}.$$

Jeżeli jednak ta relacja nie zachodzi, a więc jeżeli:

$$S_p - P_p < CT_p^{AL}.$$

to wartość dolnego sprzężenia jest niedotrzymana. Niedotrzymanie wartości dolnej sprzężenia skutkuje naliczeniem kar w wysokości iloczynu parametru CW_p^{AL} oraz stopnia niedotrzymania sprzężenia określanego jako CK_p^{AL}.

Parametr CT_p^{AU} określa maksymalną wartość, o jaką mogą różnić się charakterystyki $S_{i,j}$ oraz $P_{k,l}$. Powinna więc zachodzić relacja:

$$S_p - P_p \leq CT_p^{AU}.$$

Jeżeli powyższa relacja nie zachodzi, a więc jeżeli:

$$S_p - P_p > CT_p^{AU},$$

to wartość górnego sprzężenia jest niedotrzymana. Niedotrzymanie wartości dolnej sprzężenia skutkuje naliczeniem kar w wysokości iloczynu parametru CW_p^{AU} oraz stopnia niedotrzymania sprzężenia określanego jako CK_p^{AU}.

Pomiędzy parametrami musi zachodzić zależność:

$$CT_p^{AL} \leq CT_p^{AU}.$$

Przykład, na którym można zaprezentować zasadę działania dodatkowych elastycznych sprzężeń czasowych, jest analogiczny do tego przedstawionego w tabeli 9.

Zdefiniowanie dodatkowych elastycznych sprzężeń czasowych pozwoli na utworzenie optymalnego harmonogramu realizacji przedsięwzięć wieloobiektowych w sposób elastyczny. Opracowane sprzężenia zostaną wykorzystane w czasowo-kosztowym modelu priorytetowego harmonogramowania.

3.3. Opis opracowanej metody

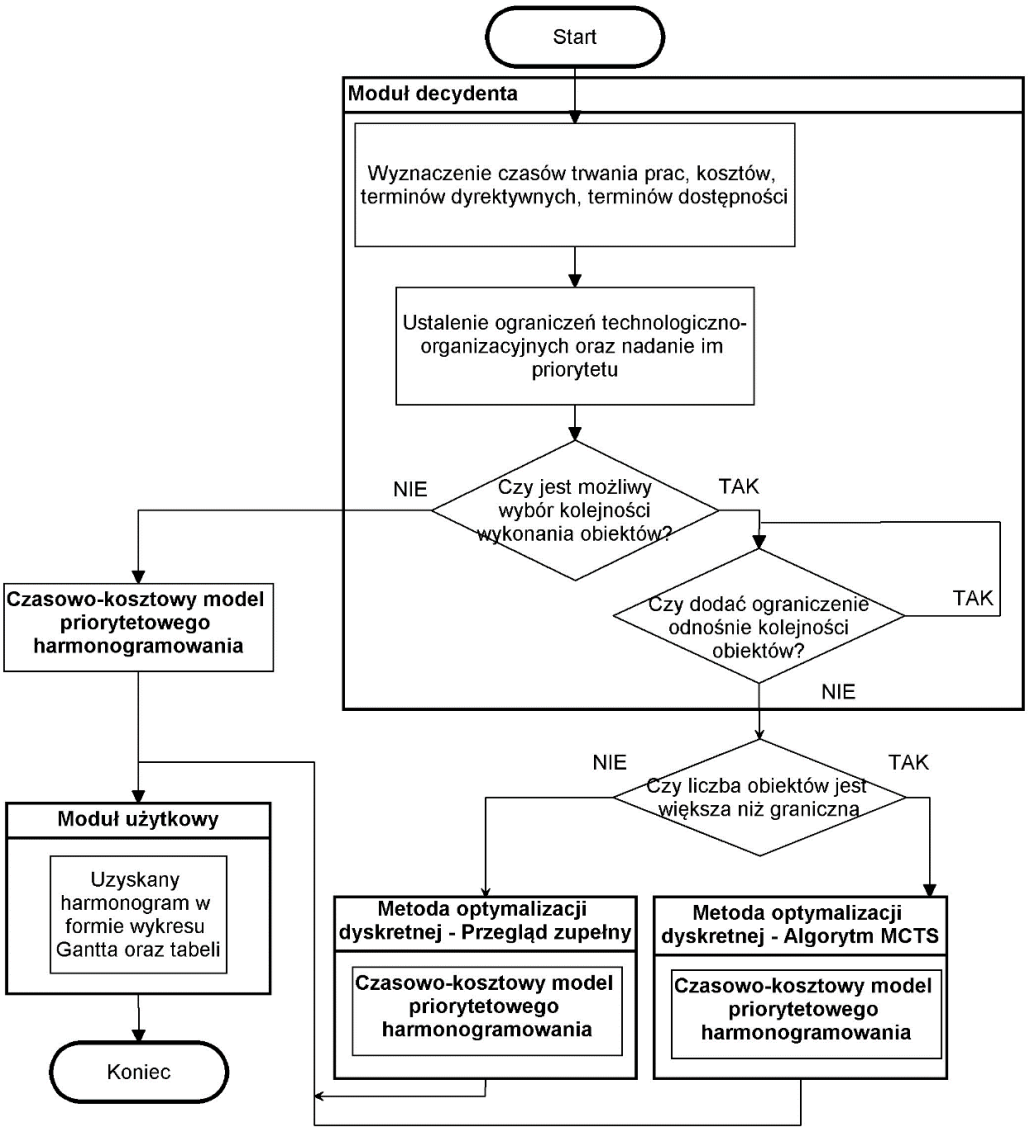
Na rysunku 17 przedstawiono schemat opracowanej metody priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych, którą można zastosować do optymalizacji czasowo-kosztowej przy określonych w sposób elastyczny ograniczeniach organizacyjno-technologicznych. W skład metody wchodzi kilka modułów i modelów:

- moduł decydenta;
- czasowo-kosztowy model harmonogramowania priorytetowego;
- metoda optymalizacji dyskretnej uszeregowania obiektów;
- moduł użytkowy.

Pierwszym etapem opracowanej metody jest przygotowanie danych przez decydenta. Decydent powinien wyznaczyć lub uzyskać odpowiednie dane liczbowe, które w sposób jednoznaczny ustalą parametry zadania optymalizacyjnego. Dodatkowo decydent powinien ustalić ograniczenia organizacyjno-technologiczne. Ograniczenia te powinny odzwierciedlać oczekiwania decydenta w stosunku do organizacji pracy brygad na budowie oraz zależności technologiczne, jakie będą występować w rzeczywistości na budowie. Dokładny sposób określenia ograniczeń przedstawiono w rozdziale 3.3.1. Jeżeli decydent nie ma wpływu na kolejność realizacji obiektów, kolejny krok stanowi wyznaczenie optymalnego harmonogramu z wykorzystaniem czasowo-kosztowego modelu priorytetowego harmonogramowania opisanego w rozdziale 3.3.2. Jeżeli jednak decydent ma wpływ na kolejność obiektów, musi on określić narzucone (np. przez inwestora) lub technologiczne ograniczenia związane z kolejnością wykonania obiektów. Jeżeli decydent może wpłynąć na kolejność wykonania obiektów, istnieje możliwość optymalizacji tej kolejności. Optymalizacja kolejności realizacji obiektów zostanie zrealizowana przez moduł optymalizacji dyskretnej uszeregowania obiektów. W zależności od liczby obiektów, jakie są do zrealizowania, należy zastosować odmienne metody optymalizacji. Jeżeli liczba obiektów jest mniejsza lub równa od granicznej liczby obiektów ($\boldsymbol{n} \leq \widehat{\boldsymbol{n}}$), to należy przeprowadzić przegląd zupełny. W przeciwnym wypadku należy zastosować inną metodę optymalizacji dyskretnej. Dla realizacji składającej się z dużej ($\boldsymbol{n} > \widehat{\boldsymbol{n}}$) liczby obiektów zostanie zastosowana zmodyfikowana metoda Monte Carlo Tree Search (MCTS). Wybór wartości liczby $\widehat{\boldsymbol{n}}$ jest zagadnieniem skomplikowanym, zależy od mocy obliczeniowej komputera oraz od czasu, jaki można przeznaczyć na obliczenia. Autor zaleca, aby wartość $\widehat{\boldsymbol{n}}$ nie przekraczała 10. W ramach zmodyfikowanej metody MCTS, należy określić ograniczenia narzucone na kolejność realizacji obiektów. Metoda optymalizacji dyskretnej została omówiona w rozdziale 3.3.4. Ostatni etap metody jest realizowany przez moduł użytkowy. Moduł ten pozwala na przedstawienie

uzyskanego rozwiązania w postaci harmonogramu oraz w postaci tabeli zawierającej terminy rozpoczęcia i zakończenia procesów. Możliwości opracowanego modułu użytkowego przedstawiono w rozdziale 3.3.5. Zaprezentowany algorytm został w całości zaimplementowany w języku programowania Python.

Wszystkie części opracowanej metody dokładnie opisano w kolejnych rozdziałach.



Rysunek 17: Schemat metody priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych

3.3.1. Moduł decydenta

Decydent (przedstawiciel generalnego wykonawcy – zazwyczaj kierownik budowy) musi zebrać lub opracować informacje kluczowe dla działania metody. Wszystkie dane powinny być określone w sposób deterministyczny. Nie wszystkie jednak muszą być uwzględnione (poza czasem trwania procesu). Dane jakie należy wprowadzić do opracowanej metody to:

- **czas trwania poszczególnych procesów wykonywanych przez poszczególne brygady na poszczególnych obiektach.** Dane takie można uzyskać na dwa podstawowe sposoby. Pierwszy to skorzystanie z Katalogów Nakładów Rzeczowych (KNR) i ustalenie dla danego przedmiaru czasu trwania procesu (dla ustalonej wielkości brygady). Jest to jednak metoda niedokładna. Decydent może nie mieć informacji na temat liczebności odpowiednich brygad, jego obliczenia należy więc traktować jako przybliżone. Drugą metodą jest ustalenie czasu wykonania procesu na poszczególnych obiektach z wybraną brygadą (którą można utożsamić z podwykonawcą). Jest to metoda dokładniejsza, pozwalająca potraktować deklarację podwykonawcy jako wiążącą oraz uwzględnić ją podczas podpisywania umowy. Dodatkowo decydent ma możliwość skracania czasu trwania poszczególnych procesów. Przyspieszenie czasu wykonania niektórych procesów jest częstą praktyką, z której korzystają kierownicy robót. Skrócenie czasu trwania procesów można uzyskać poprzez zatrudnienie dodatkowych ludzi, przeznaczenie dodatkowych maszyn, zmianę organizacji pracy lub zmianę technologii wykonania prac Sposób, w jaki można dokonywać skrócenia poszczególnych procesów nie będzie analizowany w rozprawie. Skrócenie czasu realizacji procesów pozwala na dotrzymanie terminów narzuconych przez inwestora lub otworzenie frontów robót dla innych brygad. Szczegóły na temat skracania czasu trwania procesów można znaleźć w rozdziale 3.3.3;
- **koszt bezpośredni wykonania wszystkich procesów.** Dane tego typu również można pozyskać na różne sposoby. Uwzględnienie informacji zawartych w kosztorysie (koszty robocizny, materiałów i sprzętu) (Plebankiewicz, 2007) jest metodą szacunkową. Jednak dużo dokładniejsze dane uzyskuje się podczas ustalenia warunków umowy z podwykonawcami. To podwykonawca ustala koszty danej roboty według własnych procedur. Decydent powinien również oszacować koszty przyspieszenia realizacji poszczególnych procesów;
- **koszty pośrednie.** W skład kosztów pośrednich wchodzą: płace stałego personelu budowy, koszty montażu i demontażu zaplecza tymczasowego, koszty narzędzi i drobnego sprzętu, koszty BHP, koszty zużycia materiałów oraz energii na cele administracyjne, koszty eksploatacji służbowych samochodów osobowych itp. Koszty pośrednie można wyznaczyć metodą preliminarzową lub metodą wskaźnikową. Decydent powinien przyjąć wartość jednostkowych kosztów pośrednich. Jest to uproszczenie, ponieważ wysokość kosztów pośrednich jest

inna na różnych etapach realizacji. Przyśpiesza to jednak zdecydowanie obliczenia i nie zaburza liniowości opracowywanych modelów;

- **kary umowne za niedotrzymanie terminów dyrektywnych oraz premie za wcześniejsze wykonanie obiektów.** Wartości te wynikają bezpośrednio z umowy zawartej pomiędzy generalnym wykonawcą a inwestorem. Możliwość naliczenia opłat za niedotrzymanie terminu dyrektywnego wynika z art. 483 i 484 kodeksu cywilnego. Kodeks cywilny (art. 476) rozróżnia zwłokę (niedotrzymanie terminu niewynikające z winy generalnego wykonawcy) oraz opóźnienie (niedotrzymanie terminu wynikające z winy generalnego wykonawcy). W rozprawie każde niedotrzymanie terminu oznacza winę generalnego wykonawcy. Z reguły wartość kar, jakie należy zapłacić za niedotrzymanie terminu wynoszą od 0,05% do nawet 1% wartości umowy za każdy dzień zwłoki. Jednak za wcześniejsze wykonanie obiektów inwestor może wypłacić premię dla generalnego wykonawcy. Premia ta może być określona w umowie pomiędzy inwestorem a generalnym wykonawcą. Wypłacanie premii za wcześniejsze wykonanie danego obiektu jest zasadne, ponieważ umożliwia to inwestorowi wcześniejsze użytkowanie tegoż obiektu i czerpanie z tego tytułu zysków (Hegazy & Wassef, 2001);
- **kary umowne za niedotrzymanie ciągłości pracy brygad**. Poszczególne brygady (podwykonawcy) mogą zażądać dostępności frontów roboczych podczas ich pracy przy realizacji wszystkich obiektów. Żądanie to może mieć odzwierciedlenie w umowie pomiędzy generalnym wykonawcą a podwykonawcą. Wysokość takich kar za każdy dzień przestoju powinna odpowiadać kosztom pracy ludzi i sprzętu w ciągu jednego dnia pracy danego podwykonawcy;
- **ograniczenia dostępności brygad i obiektów.** Decydent powinien posiadać informacje na temat terminów dostępności poszczególnych brygad oraz terminów dostępności poszczególnych obiektów. Wynikają one bezpośrednio z ustaleń generalnego wykonawcy z podwykonawcami oraz inwestorem. Ograniczenia dla każdego obiektu i każdej brygady są określone przez termin najwcześniejszy rozpoczęcia procesu oraz termin najpóźniejszy zakończenia procesu. Ograniczenia te muszą być bezwzględnie dotrzymane;
- **ograniczenia technologiczne i organizacyjne.** Decydent powinien mieć ustalone wszystkie ograniczenia organizacyjno-technologiczne, które należy dotrzymać podczas realizacji przedsięwzięcia wieloobiektowego. Ograniczenia takie decydent sformułuje w postaci rankingu ograniczeń. Problem uwzględnienia

ograniczeń technologiczno-organizacyjnych w sposób elastyczny został opisany w rozdziale 3.2;

- **ograniczenia narzucone na kolejność wykonania obiektów.** W przypadku możliwości zmiany kolejności wykonania obiektów decydent powinien wprowadzić reguły, jakimi należy się kierować przy jej ustalaniu. W przypadku braku możliwości zmiany kolejności wykonania obiektów, w wyniku np. narzucenia kolejności przez inwestora, nie zostanie zastosowana w procesie optymalizacji metoda optymalizacji dyskretnej. Wstępne założenia związane z tym zagadnieniem zostały omówione w rozdziale 3.3.4.

Decydent powinien posiadać jak najwięcej opisanych powyżej informacji oraz wprowadzić je do implementacji komputerowej prezentowanej metody priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych. Im więcej dokładnych danych zostanie wprowadzonych do metody, tym wyniki będzie można uznać za bardziej wiarygodne. Poprawność wprowadzanych danych nie będzie sprawdzana i weryfikowana przez implementację komputerową. Decydent powinien mieć na uwadze zasadę GIGO (Garbage In, Garbage Out) i mieć świadomość, że to on jest odpowiedzialny za nieodpowiednie zebranie i wprowadzenie danych.

Należy zwrócić uwagę, że w metodzie analizowana jest sytuacja sprzecznych interesów stron biorących udział w wieloobiektowym przedsięwzięciu budowlanym. Każdy z uczestników przedsięwzięcia wieloobiektowego działa racjonalnie, czyli dąży do osiągnięcia maksymalnego zysku (minimalnej straty). W uproszczeniu można przyjąć, że inwestor oraz generalny wykonawca nie stanowią jednego przedsiębiorstwa, ani nie łączą ich żadne relacje partnerskie, ponieważ mogłoby to wpływać na rozliczenia finansowe pomiędzy nimi.

Niekiedy dochodzi do sytuacji, kiedy inwestor jest jednocześnie generalnym wykonawcą. W tym wypadku zastosowanie proponowanej w rozprawie metody będzie również możliwe, jednak nie należy wtedy uwzględniać np. kar za niedotrzymanie terminów dyrektywnych lub kar za niedotrzymanie dostępności brygad (chociaż można je ewentualnie uwzględnić, ale wówczas można je traktować jako tzw. koszty utraconych korzyści). W łatwy sposób można będzie modelować konieczność wykonania przedsięwzięcia w wyznaczonym terminie poprzez modyfikację parametrów kosztu niedotrzymania terminów lub modelować różne założenia decyzyjne, które będą odwzorowywać pracę na placu budowy.

Decydent ma możliwość zarządzania brygadami jedynie w ramach danego przedsięwzięcia wieloobiektowego. Nie może kierować brygad do realizacji innych przedsięwzięć oraz nie może angażować dodatkowych brygad.

3.3.2. Czasowo-kosztowy model priorytetowego harmonogramowania

Zdefiniowanie elastycznych sprzężeń czasowych pozwoliło na sformułowanie modelu programowania matematycznego, realizującego koncepcję harmonogramowania priorytetowego przy optymalizacji czasowo-kosztowej.

Niech dane będą następujące zbiory:

$\hat{O} = \{(1,1),(2,1) \dots (n-1,1),(1,2) \dots (n-1,m)\}$ – zbiór par indeksów pomocny przy określaniu sprzężeń czasowych pomiędzy procesami na różnych obiektach;

$\hat{B} = \{(1,1),(2,1) \dots (n,1),(1,2) \dots (n,m-1)\}$ - zbiór par indeksów pomocny przy określaniu sprzężeń czasowych pomiędzy procesami różnych brygad;

$\widehat{W} = \{(1,1),(2,1) \dots (n,1),(1,2) \dots (n,m)\}$ - zbiór par indeksów wszystkich procesów.

Model programowania liniowego został określony w następujący sposób:

Dane:

$$tgr_{i,j}, kgr_{i,j}, t_{i,j,s}, a_{i,j,s}, k_{koś}, kp_i, kn_i, kc_j, Td_i, T_i^{SO}, T_i^{FO}, T_j^{SB}, T_j^{FB}, C_{i,j}^{O}, C_{i,j}^{B}, C_p^{A} \tag{1}$$

gdzie:

$$C_{i,j}^{O} = (CT_{i,j}^{OL}, CW_{i,j}^{OL}, CT_{i,j}^{OU}, CW_{i,j}^{OU}) \tag{2}$$

$$C_{i,j}^{B} = (CT_{i,j}^{BL}, CW_{i,j}^{BL}, CT_{i,j}^{BU}, CW_{i,j}^{BU}) \tag{3}$$

$$C_p^{A} = (CT_p^{AL}, CW_p^{AL}, CT_p^{AU}, CW_p^{AU}, S_p, P_p) \tag{4}$$

Zmienne decyzyjne:

$$\begin{gathered} t_{i,j}, d_{i,j,s}, NWR_{i,j}, NWZ_{i,j}, NPR_{i,j}, NPZ_{i,j}, ZC_{i,j}, \\ CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU}, CK_p^{AL}, CK_p^{AU}, p_i, n_i \end{gathered} \tag{5}$$

Funkcja celu:

$$\overline{K(\pi)} = K_{bez} + K_{poś} + K_p - K_n + K_c + K_{CO} + K_{CB} + K_{CA} + \mathrm{KT} \rightarrow min \tag{6}$$

gdzie:

$$K_{bez} = \sum_{i=1}^{n} \sum_{j=1}^{m} \left(\sum_{k=1}^{s} d_{i,j,\mathrm{k}} \cdot a_{i,j,s} + kgr_{i,j} \right) \tag{7}$$

$$K_{poś} = NWZ_{n,m} \cdot k_{indir} \tag{8}$$

$$K_p = \sum_{i=1}^{n} p_i \cdot kp_i \tag{9}$$

$$K_n = \sum_{i=1}^{n} n_i \cdot kn_i \tag{10}$$

$$K_c = \sum_{j=1}^{m} \left(NWZ_{n,j} - NWR_{1,j} - \sum_{i=1}^{n} t_{i,j} \right) kc_j \tag{11}$$

$$K_{CO} = \sum_{i=1}^{n} \left(CW_{i,j}^{OL} \cdot CK_{i,j}^{OL} + CW_{i,j}^{OU} \cdot CK_{i,j}^{OU} \right) \tag{12}$$

$$K_{CB} = \sum_{j=1}^{m} \left(CW_{i,j}^{BL} \cdot CK_{i,j}^{BL} + CW_{i,j}^{BU} \cdot CK_{i,j}^{BU} \right) \tag{13}$$

$$K_{AB} = \sum_{p=1}^{u} \left(CW_{p}^{AL} \cdot CK_{p}^{AL} + CW_{p}^{AU} \cdot CK_{p}^{AU} \right) \tag{14}$$

$$KT = \sum_{(i,j) \in \widetilde{W}} \left(\left(NWZ_{i,j} - NPZ_{i,j} \right) \cdot \rho \right), \qquad \text{gdzie } \rho \ll 1 \tag{15}$$

Ograniczenia:

$$t_{i,j} \geq tgr_{i,j} \tag{16}$$

$$t_{i,j} \leq tgr_{i,j} + \sum_{k=1}^{s} t_{i,j,k} \tag{17}$$

$$d_{i,j,k} \leq t_{i,j,k} \tag{18}$$

$$\bigwedge_{s \in \{1,2 \ldots z\}} tgr_{i,j} + \sum_{k=1}^{s} d_{i,j,k} \leq t_{i,j} \tag{19}$$

$$NWZ_{i,j} = NWR_{i,j} + t_{i,j} \tag{20}$$

$$NPZ_{i,j} = NPR_{i,j} + t_{i,j} \tag{21}$$

$$ZC_{i,j} = NPZ_{i,j} - NWZ_{i,j} \tag{22}$$

$$NWR_{i+1,j} \geq NWZ_{i,j} + CT_{i,j}^{OL} - CK_{i,j}^{OL} \tag{23}$$

$$NWR_{i+1,j} \leq NWZ_{i,j} + CT_{i,j}^{OU} + CK_{i,j}^{OU} \tag{24}$$

$$NPR_{i+1,j} \geq NPZ_{i,j} + CT_{i,j}^{OL} - CK_{i,j}^{OL} \tag{25}$$

$$NPR_{i+1,j} \leq NPZ_{i,j} + CT_{i,j}^{OU} + CK_{i,j}^{OU} \tag{26}$$

$$NWR_{i,j+1} \geq NWZ_{i,j} + CT_{i,j}^{BL} - CK_{i,j}^{BL} \tag{27}$$

$$NWR_{i,j+1} \leq NWZ_{i,j} + CT_{i,j}^{BU} + CK_{i,j}^{BU} \tag{28}$$

$$NPR_{i,j+1} \geq NPZ_{i,j} + CT_{i,j}^{BL} - CK_{i,j}^{BL} \tag{29}$$

$$NPR_{i,j+1} \leq NPZ_{i,j} + CT_{i,j}^{BU} + CK_{i,j}^{BU} \tag{30}$$

$$S_p \geq P_p + CT_p^{AL} - CK_p^{AL} \tag{31}$$

$$S_p \leq P_p + CT_p^{AU} + CK_p^{AU} \tag{32}$$

$$NWZ_{i,m} - p_i + n_i = Td_i \tag{33}$$

$$NWR_{i,1} \geq T_i^{SO} \tag{34}$$

$$NPZ_{i,m} \leq T_i^{FO} \tag{35}$$

$$NWR_{1,j} \geq T_j^{SB} \tag{36}$$

$$NPZ_{n,j} \leq T_j^{FB} \tag{37}$$

$$NWZ_{n,m} = NPZ_{n,m} \tag{38}$$

$$t_{i,j}, d_{i,j,s}, NWR_{i,j}, NWZ_{i,j}, NPR_{i,j}, NPZ_{i,j}, ZC_{i,j}, CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU}, CK_p^{AL}, CK_p^{AU}, p_i, n_i \geq 0 \tag{39}$$

Aby przedstawiony model mógł zostać zastosowany, należy dysponować następującymi danymi (1):

- $tgr_{i,j}$ – czas graniczny procesu wykonywanego na obiekcie *i* przez brygadę *j*. Omówiono dokładnie w rozdziale 3.3.3;
- $kgr_{i,j}$ – koszt graniczny procesu wykonywanego na obiekcie *i* przez brygadę *j*. Omówiono dokładnie w rozdziale 3.3.3;
- $t_{i,j,k}$ – długość przedziału, na jakim obowiązuje zlinearyzowana funkcja kosztów od czasu. Omówiono dokładnie w rozdziale 3.3.3;
- $a_{i,j,k}$ – współczynnik kierunkowy zlinearyzowanej funkcji kosztów od czasu w przedziale $t_{i,j,k}$. Omówiono dokładnie w rozdziale 3.3.3;
- $k_{poś}$ – jednostkowe koszty pośrednie przedsięwzięcia wieloobiektowego. Omówiono dokładnie w rozdziale 3.3.1;
- kp_i – jednostkowe koszty za niedotrzymanie terminu dyrektywnego na obiekcie *i*;
- kn_i – jednostkowa premia za wcześniejsze wykonanie obiektu *i*;
- kc_j – jednostkowe koszty za niedotrzymanie ciągłości pracy brygady *j*;
- Td_i - termin dyrektywny zakończenia prac na obiekcie *i*;
- T_i^{SO} – najwcześniejszy termin dostępności obiektu *i*;
- T_i^{FO} – najpóźniejszy termin dostępności obiektu *i*;
- T_j^{SB} – najwcześniejszy termin dostępności brygady *j*;
- T_j^{FB} – najpóźniejszy termin dostępności brygady *j*;

Dodatkowo danymi w modelu są elastyczne sprzężenia czasowe oraz dodatkowe elastyczne sprzężenia czasowe (zostały one szerzej opisane w rozdziale 3.2):

- $C_{i,j}^{O}$ – elastyczne sprzężenia czasowe pomiędzy procesami na kolejnych obiektach wykonywanych przez tę samą brygadę, określone poprzez zestaw parametrów (2);
- $C_{i,j}^{B}$ – elastyczne sprzężenia czasowe pomiędzy procesami wykonywanymi przez kolejne brygady na jednym obiekcie, określone przez zestaw parametrów (3);
- C_{p}^{A} – dodatkowe elastyczne sprzężenia czasowe, określone przez zestaw parametrów (4).

Zmiennymi decyzyjnymi w modelu są:

- $t_{i,j}$ – czas trwania procesu (*i,j*);
- $d_{i,j,k}$ - zmienna linearyzująca zależność kosztu od czasu wykonania procesu. Omówione dokładnie w rozdziale 3.3.3;
- $NWR_{i,j}$ – najwcześniejszy termin rozpoczęcia procesu (*i,j*);
- $NWZ_{i,j}$ – najwcześniejszy termin zakończenia procesu (*i,j*);
- $NPR_{i,j}$ – najpóźniejszy termin rozpoczęcia procesu (*i,j*);
- $NPZ_{i,j}$ – najpóźniejszy termin zakończenia procesu (*i,j*);
- $ZC_{i,j}$ – całkowity zapas czasu procesu (*i,j*);
- p_i - wartość opóźnienia w stosunku do terminu dyrektywnego na obiekcie *i;*
- n_i - wartość wcześniejszego wykonania obiektu *i* w stosunku do terminu dyrektywnego;
- oraz zmienne realizujące koncepcję harmonogramowania priorytetowego. (zostały one dokładnie opisane w rozdziale 3.2): $CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU}, CK_{p}^{AL}, CK_{p}^{AU}$.

Funkcją celu (6) dla pewnego uszeregowania zbioru obiektów π jest suma kosztów bezpośrednich (K_{bez}), kosztów pośrednich ($K_{poś}$), kosztów związanych z niedotrzymaniem terminów dyrektywnych poszczególnych obiektów (K_p), premii za wcześniejsze wykonanie poszczególnych obiektów ($-K_n$), kosztów przestojów pracy brygad roboczych (K_c), kosztów niedotrzymania elastycznych sprzężeń czasowych pomiędzy procesami na kolejnych obiektach wykonywanych przez tę samą brygadę (K_{CO}), kosztów niedotrzymania elastycznych sprzężeń czasowych pomiędzy procesami wykonywanymi przez kolejne brygady na jednym obiekcie (K_{CB}) oraz kosztów niedotrzymania dodatkowych elastycznych sprzężeń czasowych (K_{CA}). Ponadto, w skład funkcji celu wchodzi czynnik odpowiedzialny za wyznaczenie terminów najwcześniejszych oraz najpóźniejszych procesów (KT). Wartość funkcji celu należy zminimalizować.

Koszty bezpośrednie (7) są sumą kosztów wszystkich procesów wykonywanych przez wszystkie brygady na wszystkich obiektach w zależności od czasu trwania danego procesu. Zamiast klasycznego podejścia z zastosowaniem metody CPM-COST został opracowany zlinearyzowany model CPM-COST, który omówiono dokładniej w rozdziale 3.3.3. Koszty pośrednie (8) zależą od wysokości kosztów pośrednich jednostkowych oraz czasu trwania całej inwestycji wieloobiektowej. Jednostkowe koszty pośrednie można oszacować metodą szczegółową lub wskaźnikową. Metody te zostały przedstawione w rozdziale 3.3.1.

Koszty związane z niedotrzymaniem terminów dyrektywnych (9) wyznaczane są jako suma iloczynów kosztu jednostkowego za niedotrzymanie terminu oraz czasu trwania tego opóźnienia dla wszystkich obiektów. Koszt jednostkowy za niedotrzymanie terminu dyrektywnego może być różny dla każdego z obiektów. Kary są określone w umowie pomiędzy inwestorem, a generalnym wykonawcą i wynoszą zwykle od 0,05% do 1% wartości umowy za każdy dzień zwłoki.

Premie za wcześniejsze zrealizowanie obiektów (10) są obliczane jako iloczyn jednostkowych premii za wcześniejsze wykonanie obiektu oraz liczby dni, które określają o ile wcześniej obiekt został zrealizowany. Premie, jeżeli są, to powinny zostać określone w umowie pomiędzy inwestorem a generalnym wykonawcą.

Koszty przestojów pracy brygad roboczych (11) obliczane są jako suma iloczynów kosztu jednostkowego za przestój w pracy brygad oraz czasu trwania przestoju dla wszystkich brygad roboczych. Czas trwania przestoju obliczany jest na podstawie najwcześniejszego terminu zakończenia pracy brygady na wszystkich obiektach, najwcześniejszego terminu rozpoczęcia pracy na pierwszym obiekcie oraz czasu trwania realizacji pracy przez brygadę na wszystkich obiektach.

Dodatkowo w funkcji celu uwzględniono koszty związane z niedotrzymaniem elastycznych sprzężeń pomiędzy procesami na kolejnych obiektach wykonywanych przez tę samą brygadę (12), elastycznych sprzężeń czasowych pomiędzy procesami wykonywanymi przez kolejne brygady na jednym obiekcie (13) oraz dodatkowych elastycznych sprzężeń czasowych (14).Wartość wpływu na funkcję celu dla wszystkich typów sprzężeń wyznacza się jako sumę iloczynu niedotrzymania wartości dolnej elastycznego sprzężenia czasowego i wartości jednostkowej za niedotrzymanie tego sprzężenia oraz iloczynu niedotrzymania wartości górnej elastycznego sprzężenia czasowego i wartości jednostkowej za niedotrzymanie tego sprzężenia.

We wzorze funkcji celu jest również składnik pozwalający wyznaczyć terminy najwcześniejsze oraz najpóźniejsze dla wszystkich procesów (15). Suma najwcześniejszych terminów rozpoczęcia wszystkich procesów pomnożona przez parametr ρ wpłynie na funkcję ze znakiem plus. Skoro funkcja celu ma być minimalizowana oznacza to, że zmienne odpowiadające za najwcześniejszy termin rozpoczęcia przyjmą rzeczywiście wartości najmniejsze. Analogicznie

zostaną wyznaczone najpóźniejsze terminy rozpoczęcia, jednak ich wpływ na funkcję celu jest ze znakiem minus. Zmienne odpowiadające najpóźniejszym terminom rozpoczęcia przyjmą więc największe możliwe wartości. Parametr ρ powinien mieć taką wartość, aby nie wpływać w znaczący sposób na funkcję celu. Wartość parametru ρ powinna zostać ustalona doświadczalnie, ale sugeruje się, by była o dwa rzędy wielkości mniejsza niż najniższa wartość kosztu jednostkowego.

Na podstawie wybranych składników funkcji celu można obliczyć koszty całkowite przedsięwzięcia. Koszty całkowite przedsięwzięcia wieloobiektowe przy uszeregowaniu π będą wyznaczane na podstawie wzoru: $K(\pi) = K_{bez} + K_{poś} + K_p - K_n + K_c$.

W modelu uwzględniono następujące ograniczenia: ograniczenia (16)-(19) realizują założenia zlinearyzowanego modelu CPM-COST. Zostało to szerzej wyjaśnione w rozdziale 3.3.3. Ograniczenia (20)-(22) pozwalają wyznaczyć najwcześniejsze i najpóźniejsze terminy zakończenia procesów oraz ich całkowite zapasy czasu. Ograniczenia (23)-(32) pozwalają wyznaczyć wartość niedotrzymania odpowiednich elastycznych sprzężeń czasowych oraz dodatkowych elastycznych sprzężeń czasowych. Ograniczenie (33) pozwala wyznaczyć wartość niedotrzymania lub wcześniejszego wykonania danego obiektu. Ograniczenia (34)-(37) spełniają założenia o dostępności każdego obiektu oraz o dostępności każdej z brygad. Ograniczenie (38) sprawia, że najwcześniejszy termin zakończenia ostatniego procesu będzie równy najpóźniejszemu terminowi zakończenia. Wszystkie zmienne decyzyjne wykorzystane w modelu przyjmują wartości nieujemne (39).

Funkcja celu oraz wszystkie ograniczenia mają charakter liniowy, jest to więc model programowania liniowego, dlatego może być rozwiązany z wykorzystaniem metody Simplex. Opracowany model został zaimplementowany w języku programowania Python z zastosowaniem biblioteki PyMathProg. Przykłady obliczeniowe prezentujące zastosowanie czasowo-kosztowego modelu priorytetowego harmonogramowania przedstawiono w rozdziale 4.

3.3.3. Zlinearyzowany model czasowo-kosztowy

W opracowanej metodzie priorytetowego harmonogramowania zakłada się możliwość skrócenia czasu trwania poszczególnych procesów realizowanych przez brygady na obiektach. Aby skrócić czas trwania danego procesu, należy ponieść dodatkowe koszty. W klasycznej metodzie CPM-COST zależność czasowo-kosztowa jest liniowa, nie odpowiada to jednak rzeczywistości. Zależność tę lepiej opisywałaby funkcja nieliniowa ciągła, ponieważ im bardziej należy skrócić czas trwania procesu tym większe koszty jednostkowe należy ponieść. Z tego powodu został opracowany zlinearyzowany model czasowo-kosztowy, który może być

zastosowany nie tylko do harmonogramowania wieloobiektowych przedsięwzięć budowlanych, ale także do harmonogramowania każdego przedsięwzięcia przedstawionego w postaci MSP. Zaletą opracowanego modelu (który jest częścią czasowo-kosztowego modelu priorytetowego harmonogramowania) jest możliwość, z pewnym przybliżeniem, odwzorowania zależności nieliniowej kosztu od czasu, zależnością liniową. Dokonano takiego uproszczenia modelu nieliniowego, ponieważ optymalizacyjne modele liniowe posiadają dużo szybsze metody rozwiązania w porównaniu z modelami nieliniowymi, a szybkość jest kluczowa w kontekście opracowanej metody.

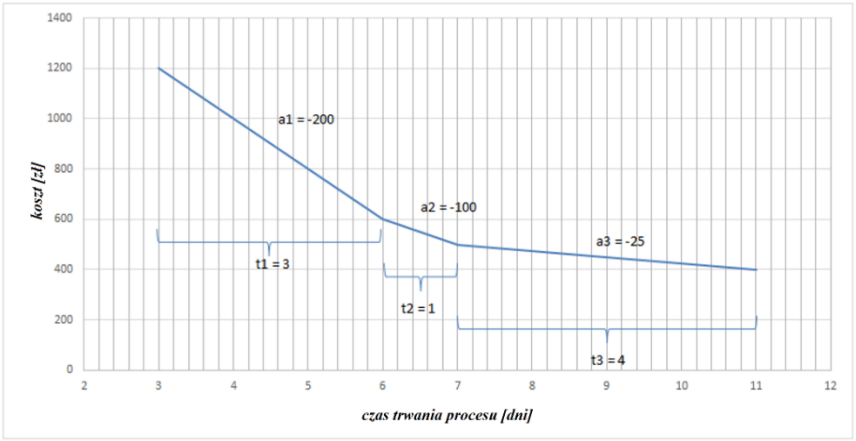
Każdy proces w sieci jest określony poprzez parametry: $tgr, kgr, t_1, a_1, t_2, a_2 \dots t_s, a_s$. Parametr tgr określa graniczny (minimalny) czas realizacji procesu, parametr kgr określa graniczne (maksymalne) koszty, jakie należy ponieść, aby wykonać proces w czasie granicznym. Parametr t_i określa długość przedziału, na jakim obowiązuje funkcja liniowa malejąca, której współczynnik kierunkowy wynosi a_i. Wartość a_i musi przyjmować wartości ujemne oraz w przypadku, gdy *s*>1, to współczynniki a_i dla kolejnych przedziałów muszą spełniać warunek przedstawiony we wzorze (40).

$$a_{i+1} > a_i, \bigwedge_{i \in \{2 \dots s\}} \tag{40}$$

Na rysunku 18 przedstawiono przykładową zależność czasowo-kosztową każdego procesu w sieci oraz interpretację graficzną parametrów t_i oraz a_i.

tgr	3
kgr	1200

k	t	a
1	3	-200
2	1	-100
3	4	-25



Rysunek 18: Zależność czasowo-kosztowa procesu wraz z graficzną interpretacją parametrów

Proces może być wykonany w czasie t, który spełnia warunek określony we wzorze (41).

$$t \in \langle tgr, tgr + \sum_{i=1}^{s} t_i \rangle \tag{41}$$

Dodatkowo niech zbiór B określa indeksy procesów początkowych w MSP, a zbiór L indeksy procesów końcowych w MSP. Dla takich założeń opracowany został model programowania matematycznego:

Dane:

$$tgr_k, kgr_k, t_{k,1}, a_{k,1}, t_{k,2}, a_{k,2} \dots t_{k,s}, a_{k,s}, Td \tag{42}$$

Zmienne decyzyjne:

$$t_k, TZ_k, TZ, d_{k,i} \text{ gdzie: } i \in \{1,2 \dots s_k\}, k \in \{1,2 \dots r\} \tag{43}$$

Funkcja celu:

$$K = \sum_{k=1}^{r} \left(\sum_{i=1}^{s_k} d_{k,i} \cdot a_{k,i} + kgr_k \right) \tag{44}$$

Ograniczenia:

$$t_k \geq tgr_k \tag{45}$$

$$t_k \leq tgr_k + \sum_{i=1}^{s_k} t_{k,i} \tag{46}$$

$$d_{k,i} \leq t_{k,i} \tag{47}$$

$$tgr_k + \sum_{i=1}^{r} d_{k,i} \leq t_k, \bigwedge_{r \in \{1,2 \dots s_k\}} \tag{48}$$

$$TZ_b \geq t_b, b \in B \tag{49}$$

$$TZ_k - TZ_p \geq t_k, \bigwedge_{p \in P_k} \tag{50}$$

$$TZ \geq TZ_l, l \in L \tag{51}$$

$$TZ = Td \tag{52}$$

$$t_k, TZ_k, Td, d_{k,i} \geq 0 \tag{53}$$

Zmiennymi decyzyjnymi w modelu są: t_k – czas trwania procesu *k*, TZ_k – termin zakończenia procesu *k*, TZ – termin zakończenia całego przedsięwzięcia, $d_{k,i}$ – zmienne linearyzujące zależność kosztu od czasu dla procesu *k*.

Funkcją celu (44) jest suma kosztów bezpośrednich wykonania wszystkich procesów. Wartość funkcji celu jest minimalizowana.

Czas trwania każdego z procesów jest większy lub równy od czasu granicznego (45) oraz mniejszy lub równy od czasu maksymalnego (46) zgodnie z zależnością opisaną we wzorze (41). Zmienne pomocnicze $d_{k,i}$ są ograniczone od góry przez odpowiednie długości przedziałów zlinearyzowanej funkcji (47). Zależność opisana we wzorze (48) pozwala wyznaczyć wartości

zmiennych $d_{k,i}$. Kolejne trzy ograniczenia określają relacje w sieci CPM. Wzór (49) określa możliwe terminy zakończenia procesów początkowych w sieci. Wzór (50) określa zależność terminów zakończenia procesów połączonych relacją zakończenie-rozpoczęcie. Wzór (51) pozwala wyznaczyć termin zakończenia całego przedsięwzięcia (oznaczonego jako TZ). Termin ten musi być równy założonemu terminowi dyrektywnemu (52). Wszystkie zmienne ($t_k, TZ_k, Td, d_{k,i}$) przyjmują wartości nieujemne (53).

Przykład zastosowania zlinearyzowanego modelu czasowo-kosztowego przedstawiono w rozdziale 4.1.

3.3.4. Metoda optymalizacji dyskretnej uszeregowania obiektów

W opracowanej metodzie priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych istnieje możliwość optymalizacji kolejności wykonania obiektów.

Należy zwrócić uwagę, że gdyby wszystkie, poza czasem trwania, parametry opracowanej metody ustalić na wartość „0", to sprowadziłaby się ona do zagadnienia szeregowania zadań typu FlowShop z funkcją celu C_{max} (Smutnicki, 2012). Zagadnienie FlowShop jest analizowany nie tylko w budownictwie, ale też w informatyce, elektronice, logistyce i innych (Li i in., 2022), (Meng & Pan, 2021), (Pempera i in., 2021), (Lin & Ying, 2016). FlowShop jest klasyfikowany jako problem NP-trudny, czyli taki, w którym niemożliwe jest znalezienie rozwiązania optymalnego w czasie wielomianowym (Karp, 1972).

Dla wybranych zagadnień szeregowania zadań powstały algorytmy o wielomianowej złożoności obliczeniowej. Są to tak zwane metody dokładne. Do przykładów takich metod można zaliczyć:

- metodę podziału i ograniczeń (Land & Doig, 1960);
- algorytm Johnsona (Johnson, 1977) i modyfikacje (Pettie, 2002);
- programowanie liniowe;
- programowanie liniowo-binarne;
- i inne.

Należy zwrócić uwagę, że dla zadań o dużych rozmiarach zastosowanie algorytmów dokładnych jest nieefektywne ze względu na czas obliczeń. Dlatego poszukiwane są metody rozwiązań przybliżonych. W miarę rosnącej mocy obliczeniowej i wszechobecnej informatyzacji, rozwiązywanie problemów NP-trudnych rozwija się w kierunku metaheurystycznych algorytmów optymalizacji dyskretnej, takich jak:

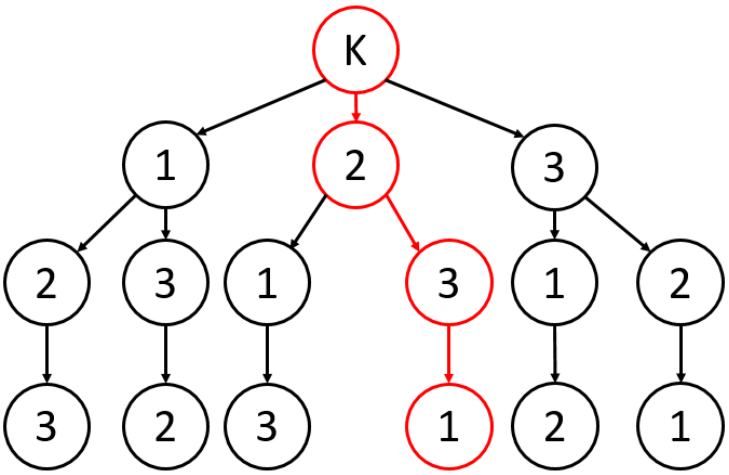
- algorytm symulowanego wyżarzania (Kirkpatrick i in., 1983);
- algorytmy genetyczne (Holland, 1992);

- algorytmy memetyczne (Yew-Soon Ong i in., 2006);
- algorytmy stada (Reynolds, 1987), (Kennedy & Eberhart, 1995);
- i inne.

Metody metaheurystyczne nie dają gwarancji uzyskania rozwiązania optymalnego, jednak uzyskiwane dzięki tym metodom rozwiązanie suboptymalne jest znacząco lepsze od losowego rozwiązania.

Ze względu na NP-trudny charakter opracowanej metody harmonogramowania przedsięwzięć wieloobiektowych wytypowano właśnie przedstawiciela metod metaheurystycznych. Po wstępnej analizie wybrana została metoda Monte Carlo Tree Search. Jest ona szczególnie popularna w zagadnieniu wyboru ruchów w grach logicznych (np. Go (Schaefers & Platzner, 2015)), została jednak zastosowana również do rozwiązywania problemów optymalizacji kombinatorycznej (Sabar & Kendall, 2015)(Jooken i in., 2023) oraz w rozwiązywaniu problemów szeregowania zadań (Lu i in., 2016)(Lubosch i in., 2018), uzyskując bardzo dobre wyniki w porównaniu z innymi metodami.

MCTS opiera się o przeszukiwanie drzewa. Drzewo to graf, w którym istnieje dokładnie jedna ścieżka łącząca korzeń z każdym pozostałym wierzchołkiem. Korzeniem jest nazywany wierzchołek, który nie ma rodziców (wierzchołek początkowy). Liściem nazywany jest taki wierzchołek w drzewie, który nie jest rodzicem (wierzchołek końcowy). Każdy wierzchołek symbolizuje podjętą decyzję. Na rysunku 19 przedstawiono przykładowe drzewo, które symbolizuje realizację składającą się z trzech obiektów. Ścieżka łącząca korzeń (oznaczony literą K) z liśćmi w sposób jednoznaczny reprezentuje uszeregowanie obiektów. Na rysunku zaznaczono kolorem czerwonym ścieżkę odpowiadającą uszeregowaniu O2, O3, O1.



Rysunek 19. Drzewo opisujące możliwe uszeregowania trzech obiektów

Metoda MCTS składa się z czterech etapów (zostały one przedstawione symbolicznie na rysunku 20):

1. Wybór (Selection)

   Pierwszym etapem metody MCTS jest procedura wyboru wierzchołków. Wierzchołki nie są wybierane w sposób losowy, lecz z zastosowaniem wzorów UCT (Upper Confidence Bound). Wzory te równoważą eksploatację i eksplorację.

2. Rozrost (Expantion)

   Po osiągnięciu wierzchołka, w którym nie ma możliwości selekcji, jest dodawany kolejny wierzchołek w sposób losowy spośród dostępnych możliwych wyborów.

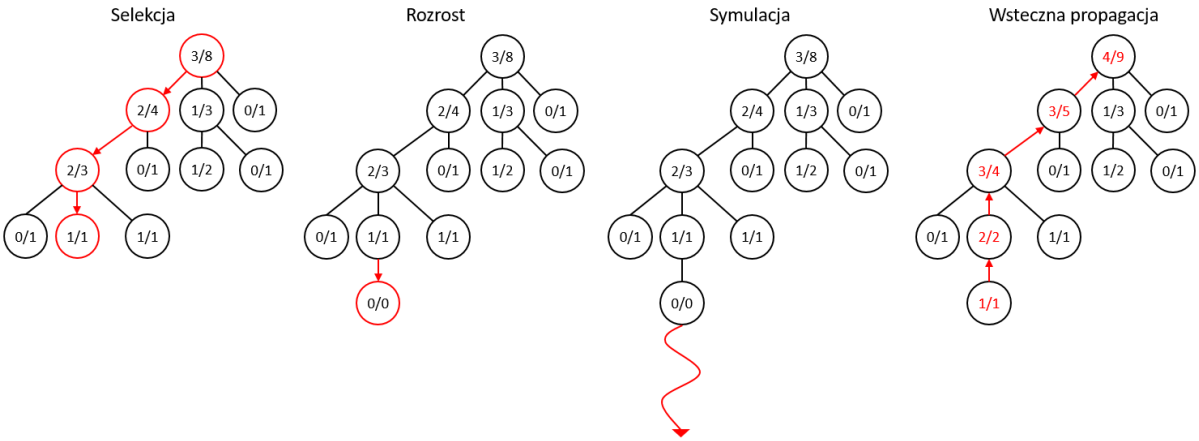
3. Symulacja (Simulation)

   Z osiągniętego w procedurze rozrostu wierzchołka kolejne wybory są dokonywane w sposób losowy. Prawdopodobieństwo każdego wyboru jest takie samo. Są dokonywane tylko takie wybory, które są zgodne z narzuconymi ograniczeniami. Symulacja prowadzi do ustalenia jednoznacznego uszeregowania.

4. Propagacja wsteczna (Backpropagation)

   Po zakończeniu symulacji należy zaktualizować wartości wierzchołków w zależności od osiągniętego wyniku.

Podstawową zaletą metody MCTS jest to, że analizuje ona wyłącznie takie rozwiązania, które są możliwe do osiągnięcia przy założonych ograniczeniach – przeszukuje wyłącznie rozwiązania dopuszczalne. Jest to duża oszczędność mocy obliczeniowej.



Rysunek 20. Procedura metody MCTS

Wyznaczenie wartości parametrów oraz sposobu wyboru wierzchołków został opisany w rozdziale 4.3.

W ramach modułu optymalizacji została opracowana modyfikacja metody, która umożliwia określanie ograniczeń w kwestii kolejności realizacji obiektów. Ograniczenia te mogą wynikać z warunków narzuconych przez inwestora. Bardzo często inwestor wymaga, aby pierwsze obiekty były realizowane od strony głównej drogi. Ma to zapewnić reklamę dla pozostałych wybudowanych obiektów. Dodatkowo mogą to być też ograniczenia prawne (wynikające np. z pozwolenia na budowę), organizacyjne (np. dostęp do danych części inwestycji), techniczne (np. rozpoczęcie budowy od robót mniej skomplikowanych), ekonomiczne (np. koniunktura rynku budowlanego) i inne. Narzucone ograniczenia powodują pomniejszenie zbioru rozwiązań dopuszczalnych.

W zmodyfikowanej metodzie MCTS można określić następujące ograniczenia:

1. i=k – obiekt o indeksie *i* ma być zrealizowany *k*-ty w kolejności;
2. i>>j – obiekt o indeksie *j* ma być wykonany po obiekcie o indeksie *i*;
3. i>j – obiekt o indeksie *j* ma być wykonany bezpośrednio po obiekcie o indeksie *i*;
4. i<>j – obiekty o indeksie *i* oraz *j* mają być zrealizowane jeden po drugim (albo *j* od razu po *i*, albo *i* od razu po *j*);
5. i_j_k_l… – obiekty o indeksach i,j,k,l... mają być realizowane jako jedna grupa (np. i,j,k,l…; j,i,k,l…;k,l,i,j…).

Przyjęto, że czas wykonania programu realizującego optymalizację dyskretną, określa użytkownik, czas ten będzie oznaczany jako T_{MCTS}. W tym założonym czasie algorytm przeszukuje przestrzeń dopuszczalną przy uwzględnieniu powyższych ograniczeń. Kod realizujący opracowaną zmodyfikowaną metodę MCTS został dołączony jako załącznik 5.

3.3.5. Moduł użytkowy

Aby opracowane metody i modele mogły być wykorzystane przez osoby odpowiedzialne za harmonogramowanie wieloobiektowych przedsięwzięć budowlanych, należy je przygotować w sposób przystępny dla użytkownika. Po procesie optymalizacji utworzony harmonogram zostanie przeniesiony do programu Microsoft Excel w postaci tabeli zawierającej nazwy, terminy rozpoczęcia i zakończenia (najwcześniejsze i najpóźniejsze). Tak przygotowane tabele można skopiować do programu Microsoft Project. Program ten jest przyjazny użytkownikowi i znany w środowisku inżynierów budownictwa. Na harmonogramie przeniesionym do programu Microsoft Project decydent będzie mógł dalej pracować z wykorzystaniem funkcji, które wykraczają poza opracowane modele i koncepcje, takie jak tworzenie diagramu sieciowego, uwzględnienie dni wolnych od pracy, przypisywanie zasobów, kontrolowanie postępu robót i inne. Dodatkowo moduł użytkowy wygeneruje „roboczy” wykres Gantta uwzględniający terminy

najwcześniejsze i najpóźniejsze dla wszystkich robót. Należy zwrócić uwagę, że w module nie można uwzględnić dni wolnych od pracy – należy to zrobić na etapie obróbki harmonogramu w specjalistycznym programie do harmonogramowania.

Do przeniesienia otrzymanych wyników do programu Excel oraz wygenerowania wykresu Gantta zostanie użyty język programowania Python oraz biblioteki Plotly, Pandas, Numpy.

3.4. Opracowane zestawy wag realizujące różne założenia decyzyjne

Poza opracowaniem modelu matematycznego realizującego koncepcję priorytetowego harmonogramowania opracowano zestawy wag, które będą w sposób elastyczny realizować założenia decyzyjne. W niniejszym rozdziale przedstawiony trzy różne grupy opracowanych wag modelu realizujących różne założenia decyzyjne.

Należy zwrócić uwagę, że w programie dedykowanym dla decydenta, nie powinien on wprowadzać wartości poszczególnych wag samodzielnie. Wymaga to opracowania bardziej przystępnych opcji programu jak lista wielokrotnego wyboru, ustawianie rankingu w postaci listy, graficzne wprowadzenie sprzężeń i ustalenie ich priorytetu w postaci liczb porządkowych. Wówczas zaimplementowany w programie algorytm na podstawie zaznaczonych opcji dobrałby odpowiednie wartości parametrów.

Preferencje decydenta dotyczące znaczenia sprzężeń są określone za pomocą zbiorów: $\overline{P_1}, \overline{P_2} \dots \overline{P_k}$. Zbiór $\overline{P_1}$ określa najmniejszy priorytet, zbiór $\overline{P_k}$ największy. Każdy zbiór jest określony w sposób: $\overline{P_l} = \{p_{l,1}, p_{l,2}, \dots, p_{l,s}, \dots, p_{l,r}\}$, gdzie r określa liczbę sprzężeń o tym samym priorytecie, $p_{l,s}$ określa rodzaj sprzężenia: $p_{l,s} \in \{C_{i,j}^O, C_{i,j}^B, C_p^A\}$.

Parametry $\widetilde{L_1} \dots \widetilde{L_k}$ określają priorytet danego zbioru sprężeń. Parametr $\widetilde{L_1}$ oznacza dostatecznie dużą liczbę oraz: $\widetilde{L_1} \ll \widetilde{L_2} \ll \dots \ll \widetilde{L_k}$. Poprzez dostatecznie dużą liczbę należy rozumieć taką wartość parametru $\widetilde{L_1}$, która jest o kilka rzędów wielkości większa niż ustalone koszty oraz nagrody jednostkowe $(a_{i,j,s}, k_{indir}, kp_i, kn_i, kc_j)$. Spowoduje to, że ustalone założenia decyzyjne będą dotrzymywane w pierwszej kolejności. Proponuje się ustalanie wartości parametru $\widetilde{L_1}$ na poziomie o 2 rzędy wielkości większego niż maksimum spośród liczb $a_{i,j,s}, k_{poś}, kp_i, kn_i, kc_j$. Każdy kolejny parametr $\widetilde{L_2} \dots \widetilde{L_k}$ powinien przyjmować wartość na poziomie o 2 rzędy wielkości większy od poprzedniego parametru. Parametry te pozwolą zrealizować koncepcję harmonogramowania priorytetowego.

3.4.1. Zestawy wag uwzględniające sprzężenia elastyczne klasyczne

W tabeli 10 przedstawiono różne typy efektów, jakie można uzyskać przy uwzględnieniu jedynie elastycznych sprzężeń czasowych. W opracowanych zestawach nie zostały uwzględnione sprzężenia elastyczne dodatkowe, a więc: $C_p^A = \emptyset$. Typy efektów, jakie można osiągnąć z zastosowaniem jedynie elastycznych sprzężeń czasowych (bez sprężeń dodatkowych) będą oznaczane literą A.

Typ efektu A.1 spełnia ograniczenia ogólne MSP (w teorii sprzężeń czasowych jest to model TCM III), a więc każdy proces może rozpocząć się tylko wtedy, gdy wszystkie poprzedniki zostały zakończone. Typ efektu A.2 służy do modelowania ciągłości realizacji procesów wykonywanych przez brygady (TCM I). Typ efektu A.3 zapewnia ciągłość realizacji procesów na obiektach roboczych (TCM II). Dla typu efektu A.4 decydent określa, na których obiektach oraz dla których brygad ma być zapewniona ciągłość realizacji procesów. Numery obiektów, na których ma być zapewniona ciągłość wykonywanych procesów, są określone poprzez zbiór $\overline{\overline{O}} = \{\overline{\overline{o_1}}, \overline{\overline{o_2}}, \dots, \overline{\overline{o_k}}\}$. Zbiór $\overline{\overline{B}} = \{\overline{\overline{b_1}}, \overline{\overline{b_2}}, \dots, \overline{\overline{b_k}}\}$ zawiera numery brygad, które powinny mieć zapewnioną ciągłość realizacji procesów. Dla typu efektu A.5 dopuszczono wykonywanie prac przez brygadę na różnych obiektach z pokrywaniem się realizacji procesów maksymalnie o s dni. Dla typu efektu A.6 dopuszczano możliwość wykonywania procesów na jednym obiekcie przez dwie brygady. Realizacja procesów pokrywa się maksymalnie o s dni. Typ efektu A.7 realizuje jednocześnie ograniczenia dla modelów A.5 oraz A.6.

Typy efektów A.8a oraz A.8b realizują koncepcję priorytetowego harmonogramowania. Poprzez priorytetowe harmonogramowanie należy rozumieć takie harmonogramowanie, w którym decydent ustala priorytet (ważność) poszczególnych ograniczeń technologiczno-organizacyjnych. Typ efektu A.8a dopuszcza możliwość zazębiania się procesów na obiektach lub pracy jednej brygady na kilku obiektach. Typ efektu A.8b nie dopuszcza takiej możliwości.

W przypadku typów efektów A.1, A.2, A.3, A.5, A.6, A.7 wartość funkcji celu $\overline{K(\pi)}$ będzie w przybliżeniu równa kosztom realizacji całego przedsięwzięcia $K(\pi)$. Wynika to z faktu, że we wzorze funkcji celu jeden z czynników (15) jest odpowiedzialny za wyznaczenie terminów najwcześniejszych oraz najpóźniejszych. Wpływ tego czynnika jest znikomy w porównaniu z pozostałymi czynnikami (np. kosztami realizacji przedsięwzięcia), ponieważ wartość parametru $\rho << 1$.

Dla typów efektów A.4, A.8a oraz A.8b wartość funkcji celu może osiągać przybliżoną wartość kosztów wykonania całego przedsięwzięcia, gdy wszystkie preferencje decydenta zostaną dotrzymane. W przypadku, gdy dotrzymanie preferencji decydenta jest niemożliwe, wartość funkcji celu informuje w jakim stopniu wymagane warunki nie zostały spełnione.

Tabela 10: Opracowane zestawy wag parametrów realizujące różne efekty decyzyjne, uwzględniające elastyczne sprzężenia czasowe

Typ efektu	Wartość parametrów	Opis efektu planistycznego	Wartość funkcji celu
A.1	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$	Ograniczenia jak w metodzie CPM	$\overline{K(\pi)} \approx K(\pi)$
A.2	$CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{OU}, CT_{i,j}^{BL} = 0$	Ciągłość realizacji procesów dla brygad	$\overline{K(\pi)} \approx K(\pi)$
A.3	$CT_{i,j}^{OU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL}, CT_{i,j}^{BU} = 0$	Ciągłość realizacji procesów na obiektach	$\overline{K(\pi)} \approx K(\pi)$
A.4	$CW_{i,j}^{OL}, CW_{i,j}^{OU} = \widetilde{L_1}$, jeżeli $j \notin \bar{\bar{B}}$ $CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$, jeżeli $i \notin \bar{\bar{O}}$ $CW_{i,j}^{OL}, CW_{i,j}^{OU} = \widetilde{L_2}$, jeżeli $j \in \bar{\bar{B}}$ $CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_2}$, jeżeli $i \in \bar{\bar{O}}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_{i,j}^{OU} = \widetilde{L_1}$, jeżeli $j \notin \bar{\bar{B}}$ $CT_{i,j}^{BU} = \widetilde{L_1}$, jeżeli $i \notin \bar{\bar{O}}$ $CT_{i,j}^{OU} = 0$, jeżeli $j \in \bar{\bar{B}}$ $CT_{i,j}^{BU} = 0$, jeżeli $i \in \bar{\bar{O}}$	Ciągłość realizacji procesów na obiektach $\bar{\bar{o}} \in \bar{\bar{O}}$ oraz ciągłość realizacji procesów dla brygad $\bar{\bar{b}} \in \bar{\bar{B}}$	Jeżeli: $CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU} = 0$ to: $\overline{K(\pi)} \approx K(\pi)$ W przeciwnym wypadku: $\overline{K(\pi)}$ określa stopień dotrzymania preferencji decydenta
A.5	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL} = -s$ $CT_{i,j}^{BL} = 0$	Możliwość wykonywania procesów na różnych obiektach przez jedną brygadę, z s-dniowym nakładaniem się procesów	$\overline{K(\pi)} \approx K(\pi)$
A.6	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL} = 0$ $CT_{i,j}^{BL} = -s$	Możliwość wykonywania procesów na jednym obiekcie przez kilka brygad, z	$\overline{K(\pi)} \approx K(\pi)$

		s-dniowym nakładaniem się procesów	
A.7	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = -s$	Kombinacja modelu A.5 i A.6	$\overline{K(\pi)} \approx K(\pi)$
A.8a	$CW_{i,j}^{tL}, CW_{i,j}^{tU}, CT_{i,j}^{tU} = \widetilde{L_1}$, jeżeli $(t,i,j) \notin (\overline{P_1} \cup \overline{P_2} \cup \ldots \cup \overline{P_k})$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_{i,j}^{tU} = 0$, jeżeli $(t,i,j) \in (\overline{P_1} \cup \overline{P_2} \cup \ldots \cup \overline{P_k})$ $CW_{i,j}^{tL}, CW_{i,j}^{tU} = \widetilde{L_2}$, jeżeli $(t,i,j) \in \overline{P_1}$ $CW_{i,j}^{tL}, CW_{i,j}^{tU} = \widetilde{L_3}$, jeżeli $(t,i,j) \in \overline{P_2}$ … … … $CW_{i,j}^{tL}, CW_{i,j}^{tU} = \widetilde{L_{k+1}}$, jeżeli $(t,i,j) \in \overline{P_k}$	Priorytetowe harmonogramowanie	Jeżeli: $CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU} = 0$ to: $\overline{K(\pi)} \approx K(\pi)$ W przeciwnym wypadku: $\overline{K(\pi)}$ określa stopień dotrzymania preferencji decydenta
A.8b	$CW_{i,j}^{tL} = \widetilde{L_{k+2}}$, jeżeli $(t,i,j) \notin (\overline{P_1} \cup \overline{P_2} \cup \ldots \cup \overline{P_k})$ $CW_{i,j}^{tU}, CT_{i,j}^{tU} = \widetilde{L_1}$, jeżeli $(t,i,j) \notin (\overline{P_1} \cup \overline{P_2} \cup \ldots \cup \overline{P_k})$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_{i,j}^{tU} = 0$, jeżeli $(t,i,j) \in (\overline{P_1} \cup \overline{P_2} \cup \ldots \cup \overline{P_k})$ $CW_{i,j}^{tL}, CW_{i,j}^{tU} = \widetilde{L_2}$, jeżeli $(t,i,j) \in \overline{P_1}$ $CW_{i,j}^{tL}, CW_{i,j}^{tU} = \widetilde{L_3}$, jeżeli $(t,i,j) \in \overline{P_2}$ … … … $CW_{i,j}^{tL}, CW_{i,j}^{tU} = \widetilde{L_{k+1}}$, jeżeli $(t,i,j) \in \overline{P_k}$	Priorytetowe harmonogramowanie z uwzględnieniem braku możliwości zachodzenia realizacji procesów na siebie, zarówno na obiektach jak i w pracy brygad	Jeżeli: $CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU} = 0$ to: $\overline{K(\pi)} \approx K(\pi)$ W przeciwnym wypadku: $\overline{K(\pi)}$ określa stopień dotrzymania preferencji decydenta

Opisane powyżej zestawy wag zostały zastosowane na przykładzie obliczeniowym opisanym w rozdziale 4.2.

3.4.2. Zestawy wag uwzględniające sprzężenia elastyczne dodatkowe

W tabeli 11 przedstawiono opracowane zestawy wag parametrów, realizujące różne efekty decyzyjne, przy uwzględnieniu elastycznych dodatkowych sprzężeń czasowych. W opracowanych zestawach zostały uwzględnione sprzężenia elastyczne dodatkowe, a więc: $C_p^A \neq \emptyset$, ograniczenia wynikające z metody CPM oraz nie uwzględniono żadnych specjalnych

ograniczeń realizujących typy efektów od A.2 do A.8. Prezentowane w tym rozdziale typy efektów będą oznaczane literą B. W każdym z typów efektów B zostały uwzględnione ograniczenia takie jak w metodzie CPM (czyli takie same jak w metodzie TCM III).

Dla typu efektu B.1 zostały uwzględnione dodatkowo sprzężenia diagonalne, które łączą najwcześniejsze terminy rozpoczęcia procesów wykonywanych przez bezpośrednio poprzedzające brygady na kolejnym obiekcie $(i+1, j-1)$. Jest to więc realizacja metody TCM IV.

Dla typu efektu B.2 zostały uwzględnione dodatkowo sprzężenia odwrotne diagonalne, czyli takie, które łączą najwcześniejsze terminy rozpoczęcia procesów wykonywanych przez kolejne brygady na bezpośrednio poprzedzającym obiekcie $(i-1, j+1)$. Realizuje to więc metodę TCM V.

Typ efektu B.3 uwzględnia zerowe sprzężenia diagonalne dla terminów najwcześniejszych. To znaczy, że różnica pomiędzy charakterystyką najwcześniejszego terminu rozpoczęcia procesu, a charakterystyką najwcześniejszego rozpoczęcia procesu realizowanego przez następną brygadę na poprzednim obiekcie $(i+1, j-1)$ będzie wynosić 0. Będzie to tożsame z metodą TCM VI.

Typ efektu B.4 uwzględnia narastanie zapasu całkowitego czasu dla procesu dowolnej k-tej brygady. Jest to typ prezentujący możliwość połączenia charakterystyk procesów takich jak zapas czasu.

Typ efektu B.5 uwzględnia zerowe sprzężenia diagonalne pomiędzy charakterystyką najwcześniejszego rozpoczęcia procesu, a charakterystyką najwcześniejszego zakończenia procesu realizowanego przez następną brygadę na poprzednim obiekcie $(i+1, j-1)$. Różnica tych charakterystyk będzie wynosić 0. Jest to tożsame z innym wariantem metody TCM VI.

Tabela 11: Opracowane zestawy wag parametrów realizujące różne efekty decyzyjne, uwzględniające elastyczne dodatkowe sprzężenia czasowe

Typ efektu	Wartość parametrów	Opis efektu planistycznego	Wartość funkcji celu
B.1	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL} = 0$ $CW_p^{AL}, CW_p^{AU}, CT_p^{AU} = \widetilde{L_1}$ $S_p = NWR_{i,j}, P_p = NWR_{i+1,j-1}$	Ograniczenia jak w metodzie CPM z uwzględnieniem sprzężeń diagonalnych dla terminów najwcześniejszych	$\overline{K(\pi)} \approx K(\pi)$

B.2	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL} = 0$ $CW_p^{AL}, CW_p^{AU}, CT_p^{AU} = \widetilde{L_1}$ $S_p = NWR_{i,j}, P_p = NWR_{i-1,j+1}$	Ograniczenia jak w metodzie CPM z uwzględnieniem sprzężeń odwrotnych diagonalnych dla terminów najwcześniejszych rozpoczęcia	$\overline{K(\pi)} \approx K(\pi)$
B.3	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL}, CT_p^{AU} = 0$ $CW_p^{AL}, CW_p^{AU} = \widetilde{L_2}$ $S_p = NPR_{i,j}, P_p = NPR_{i+1,j-1}$	Ograniczenia jak w metodzie CPM z uwzględnieniem zerowych sprzężeń diagonalnych dla najpóźniejszych terminów rozpoczęcia	$\overline{K(\pi)} \approx K(\pi)$
B.4	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL} = 1$ $CW_p^{AL}, CW_p^{AU}, CT_p^{AU} = \widetilde{L_2}$ $S_k = ZC_{i+1,k}, P_k = ZC_{i,k}$	Ograniczenia jak w metodzie CPM z narastającym o 1 dzień zapasem czasu dla procesów realizowanych przez k-tą brygadę	$\overline{K(\pi)} \approx K(\pi)$
B.5	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL}, CT_p^{AU} = 0$ $CW_p^{AL}, CW_p^{AU} = \widetilde{L_2}$ $S_k = NWR_{i,j}, P_k = NWZ_{i+1,j-1}$	Ograniczenia jak w metodzie CPM z uwzględnieniem zerowych sprzężeń diagonalnych łączących najwcześniejsze terminy rozpoczęcia i zakończenia robót	$\overline{K(\pi)} \approx K(\pi)$

Opisane powyżej zestawy wag zostały zastosowane na przykładzie obliczeniowym opisanym w rozdziale 4.2.

3.4.3. Zestawy wag uwzględniające sprzężenia elastyczne niestandardowe

Zaprezentowane w rozdziałach 3.4.1 oraz 3.4.2 zestawy wag nie realizują wszystkich możliwych założeń decyzyjnych. Opracowanie wszystkich możliwych zestawów wag nie leży w zakresie tematycznym tej rozprawy. Przytoczone zestawy pozwalają wyjaśnić możliwości stosowania opracowanego modelu harmonogramowania priorytetowego. Dokonując odpowiedniego doboru wag można zrealizować każde potencjalne niestandardowe założenie decyzyjne. W tabeli 12 opisano trzy przykładowe zastosowania opracowanej koncepcji dla przedsięwzięć o $n \geq 3$ oraz $m \geq 3$. Te zestawy wag będą oznaczone literą C.

Tabela 12: Opracowane zestawy wag parametrów realizujące różne efekty decyzyjne, uwzględniające elastyczne dodatkowe sprzężenia czasowe

Typ efektu	Wartość parametrów	Opis efektu planistycznego	Wartość funkcji celu
C.1	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL}, CT_p^{AU} = 0$ $CW_p^{AL}, CW_p^{AU} = \widetilde{L_2}$ $S_1 = NWR_{2,2}, P_1 = NWR_{1,3}$ $S_2 = NWR_{2,2}, P_2 = NWR_{3,1}$	Ograniczenia jak w metodzie CPM oraz zerowe sprzężenie diagonalne dla najwcześniejszych terminów rozpoczęcia pomiędzy procesami O3 B1, O2 B2, O1 B3	$\overline{K(\pi)} \approx K(\pi)$
C.2	$CT_{i,j}^{OU}, CT_{i,j}^{BU}, CW_{i,j}^{OL}, CW_{i,j}^{OU}, CW_{i,j}^{BL}, CW_{i,j}^{BU} = \widetilde{L_1}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_p^{AL}, CT_p^{AU} = 0$ $CW_p^{AL}, CW_p^{AU} = \widetilde{L_2}$ $S_1 = NWZ_{2,2}, P_1 = NWZ_{1,3}$ $S_2 = NWZ_{2,2}, P_2 = NWZ_{3,1}$	Ograniczenia jak w metodzie CPM oraz zerowe sprzężenie diagonalne dla najwcześniejszych terminów zakończenia pomiędzy procesami O3 B1, O2 B2, O1 B3	$\overline{K(\pi)} \approx K(\pi)$
C.3	$CW_{i,j}^{OL}, CW_{i,j}^{BL} = \widetilde{L_6}$ $CT_{i,j}^{OL}, CT_{i,j}^{BL} = 0$ $CT_{i,0}^{OU}, CT_{i,3}^{OU} = \widetilde{L_1}$ $CT_{i,1}^{OU}, CT_{i,2}^{OU} = 0$	Harmonogramowanie priorytetowe. Ograniczenia dochowane w podanej kolejności:	Jeżeli: $CK_{i,j}^{OL}, CK_{i,j}^{OU}, CK_{i,j}^{BL}, CK_{i,j}^{BU},$ $CK_{i,j}^{AL}, CK_{i,j}^{AU} = 0$ to: $\overline{K(\pi)} \approx K(\pi)$ W przeciwnym wypadku:

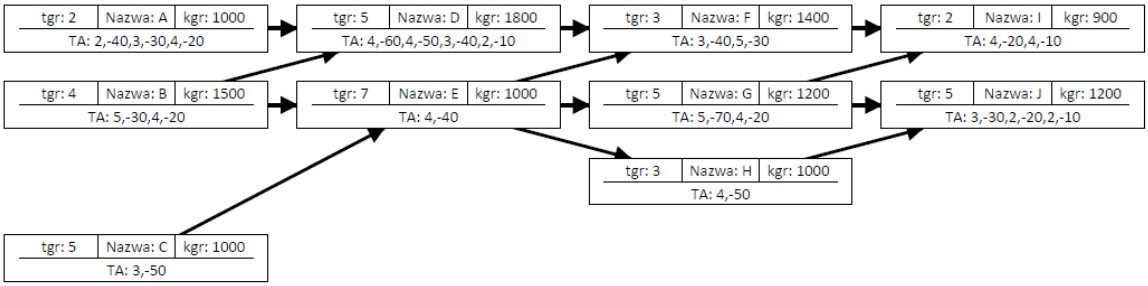
	$CW_{i,0}^{OU}, CW_{i,3}^{OU} = \widetilde{L_1}$ $CW_{i,1}^{OU} = \widetilde{L_2}$ $CW_{i,2}^{OU} = \widetilde{L_4}$ $CT_{1,j}^{BU}, CT_{3,j}^{BU} = \widetilde{L_1}$ $CT_{2,j}^{BU} = 0$ $CW_{1,j}^{BU}, CW_{3,j}^{BU} = \widetilde{L_1}$ $CW_{2,j}^{BU} = \widetilde{L_3}$ $CT_p^{AL}, CT_p^{AU} = 0$ $CW_p^{AL}, CW_p^{AU} = \widetilde{L_5}$ $S_1 = NWZ_{1,3}, P_1 = NWZ_{2,2}$ $S_2 = NWZ_{2,2}, P_1 = NWZ_{3,1}$	1) ograniczenia metody CPM 2) zerowe sprzężenia diagonalne pomiędzy najwcześniejszymi terminami zakończenia pomiędzy robotami O3 B1, O2 B2, O1 B3 3) ciągłość procesów realizowanych przez brygadę B3 4) ciągłość procesów realizowanych na obiekcie O2 5) ciągłość procesów realizowanych przez brygadę B2	$\overline{K(\pi)}$ określa stopień dotrzymania preferencji decydenta

Opisane powyżej zestawy wag zostały zastosowane na przykładzie obliczeniowym opisanym w rozdziale 4.2.

4. Przykłady obliczeniowe

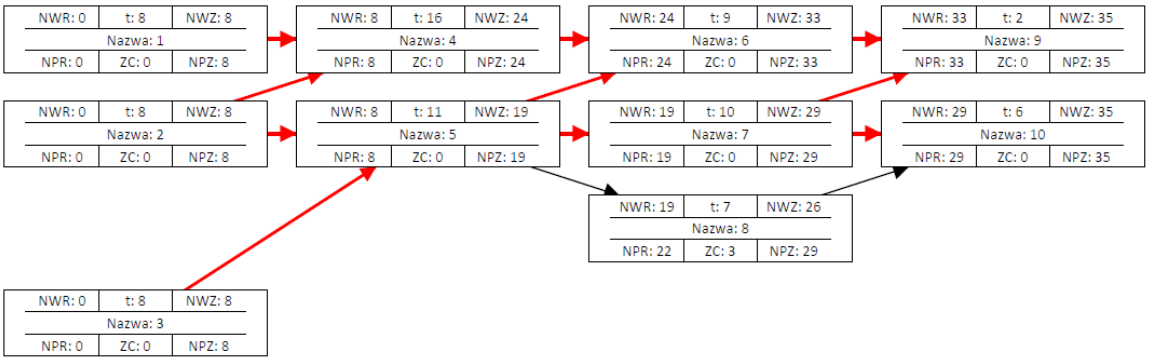
4.1. Przykład zastosowania zlinearyzowanego modelu czasowo-kosztowego

Opracowany zlinearyzowany model czasowo-kosztowy został zastosowany do obliczenia zadania optymalizacji kosztów przedsięwzięcia przy narzuconym terminie dyrektywnym. Model matematyczny przedstawiono w rozdziale 3.3.3. W skład MSP wchodzi 10 procesów. Nazwy procesów są oznaczone kolejnymi literami alfabetu. Sieć jednopunktowa została zaprezentowana na rysunku 21. Wszystkie procesy są połączone relacją FS (Finish-Start). Termin dyrektywny został założony na poziomie 35 dni.



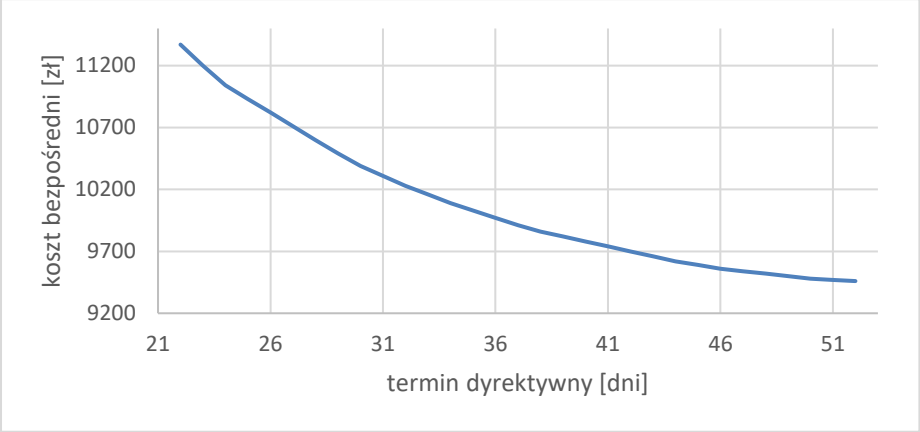
Rysunek 21: Przykładowa sieć CPM-COST przedstawiona w postaci MSP

Dla przykładowej sieci otrzymano koszty na poziomie 10030 zł. MSP dla otrzymanych czasów realizacji przedstawiono na rysunku 22. Na czerwono zaznaczono ścieżkę krytyczną.



Rysunek 22: Rozwiązanie optymalne dla terminu dyrektywnego 35

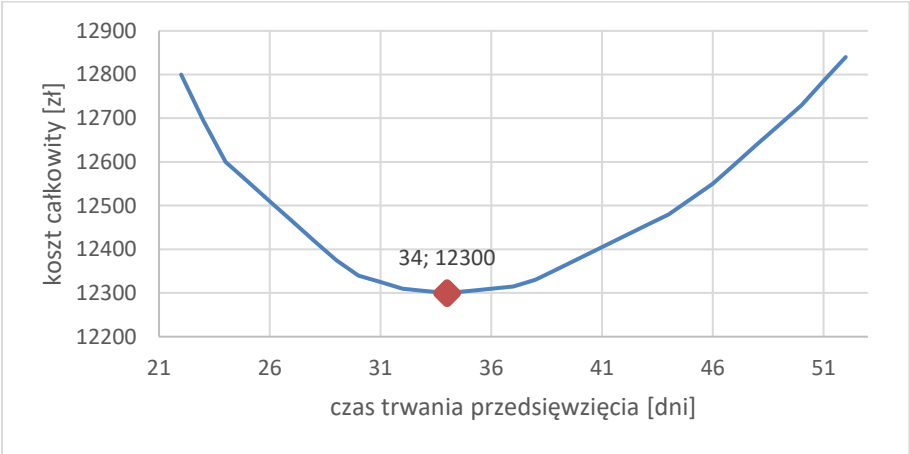
Minimalny termin dyrektywny wynosi 22 dni (przy czasach granicznych), natomiast maksymalny 52 dni (przy najdłuższych czasach trwania procesów). Obliczenia zostały wykonane dla każdego możliwego terminu dyrektywnego. Wykres kosztów w zależności od terminu dyrektywnego przedstawiono na wykresie 23. Zgodnie z oczekiwaniami, jest to funkcja malejąca, nieliniowa.



Rysunek 23: Zależność minimalnego kosztu bezpośredniego procesu od terminu dyrektywnego

Opracowany model został również zastosowany do wyznaczenia minimalnej sumy kosztów pośrednich i bezpośrednich dla zaprezentowanej wcześniej sieci. Uwzględniono dodatkowe jednostkowe koszty pośrednie $k_{poś}$ na poziomie 65zł/dzień. Dla takich danych uzyskano czas realizacji 34 dni i odpowiadające temu koszty całkowite 12300 zł.

Na rysunku 24 przedstawiono wykres kosztu całkowitego w zależności od terminu dyrektywnego. Na wykresie zaznaczono również otrzymaną minimalną wartość kosztu całkowitego.



Rysunek 24: Zależność minimalnego kosztu całkowitego od czasu trwania przedsięwzięcia

4.2. Przykład zastosowania modelu harmonogramowania priorytetowego

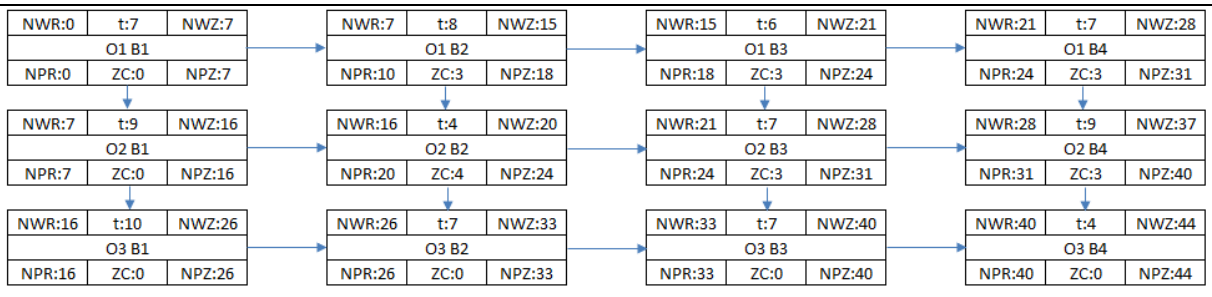
Opracowany model priorytetowego harmonogramowania został zaprezentowany na przykładzie obliczeniowym. Przyjęto fikcyjną realizację składającą się z 3 obiektów. Na każdym obiekcie mają zostać zrealizowane 4 różne rodzaje procesów. Czas trwania poszczególnych procesów przedstawiono w tabeli 13. Dla tak przygotowanych danych, opracowano różne efekty decyzyjne. Dla czytelności przykładów założono, że istnieją jedynie jednostkowe koszty pośrednie w wysokości 1 oraz brak jest dodatkowych ograniczeń (takich jak dostępność brygad lub dostępność obiektów).

Tabela 13: Czasy trwania procesów wykonywanych przez brygady Bj na obiektach Oi.

	B1	**B2**	**B3**	**B4**
O1	7	8	6	7
O2	9	4	7	9
O3	10	7	7	4

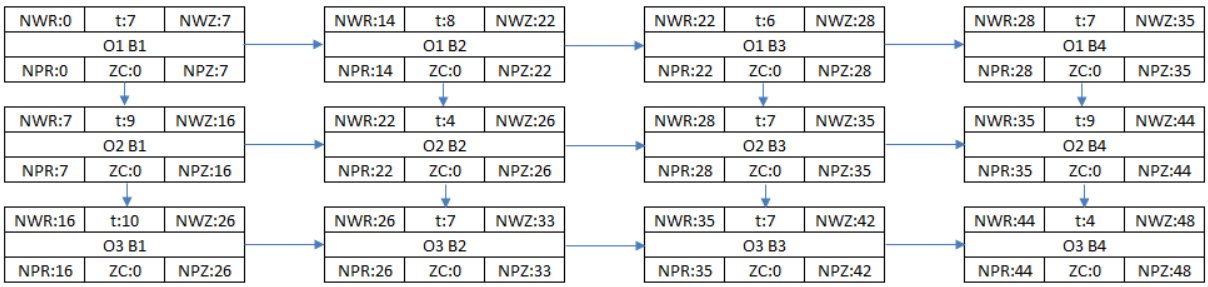
Dla powyższych danych zostały wyznaczone harmonogramy optymalne z uwzględnieniem różnych efektów decyzyjnych, opisanych w rozdziale 3.4.1. Wyniki zostały otrzymane na podstawie wykonania kodu będącego załącznikiem 3 do niniejszej rozprawy, a zaprezentowano je w tabeli 14.

Tabela 14: Otrzymane wyniki dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od A.1 do A.8

Typ efektu	**Ograniczenia**	**Wartość funkcji celu**	**Czas trwania**
A.1	Ograniczenia jak w metodzie CPM.	~43.99	44
Komentarz: Najprostszy opracowany model. Ograniczenia w tym modelu są identyczne jak w modelu CPM. Uzyskane wyniki są tożsame z wynikami uzyskanymi innymi metodami. Wartość funkcji celu obliczono w oparciu o czas trwania, jednostkowe koszty pośrednie oraz współczynnik pozwalający wyznaczyć terminy najwcześniejsze i najpóźniejsze (z bardzo małym współczynnikiem). Istnieje jedna ścieżka krytyczna.			
			

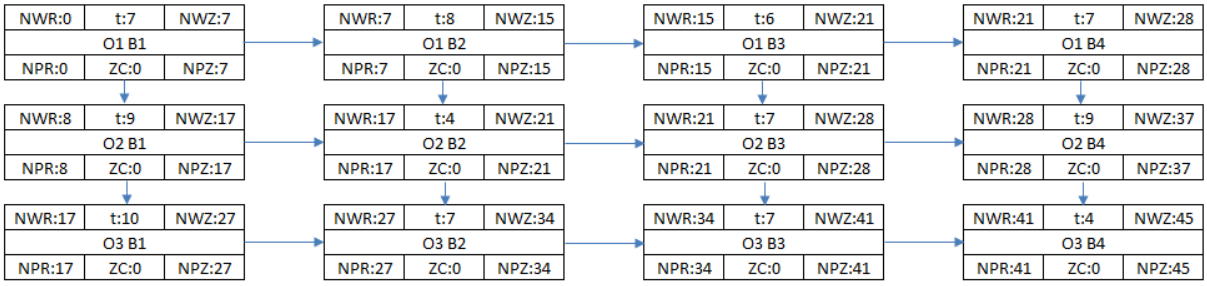
A.2	Ciągłość realizacji procesów dla brygad.	~48	48

Komentarz: Model, który pozwala wyznaczyć harmonogram zapewniający ciągłość pracy brygad. Czas realizacji takiego przedsięwzięcia jest dłuższy w stosunku do poprzedniego przykładu o 4 dni. Dodatkowo występują przerwy w pracy na obiektach. Podobne rozwiązanie uzyskamy korzystając z teorii sprzężeń czasowych. Wartość funkcji celu obliczono w oparciu o czas trwania, jednostkowe koszty pośrednie oraz współczynnik pozwalający wyznaczyć terminy najwcześniejsze i najpóźniejsze (z bardzo małym współczynnikiem). Wszystkie procesy są krytyczne.



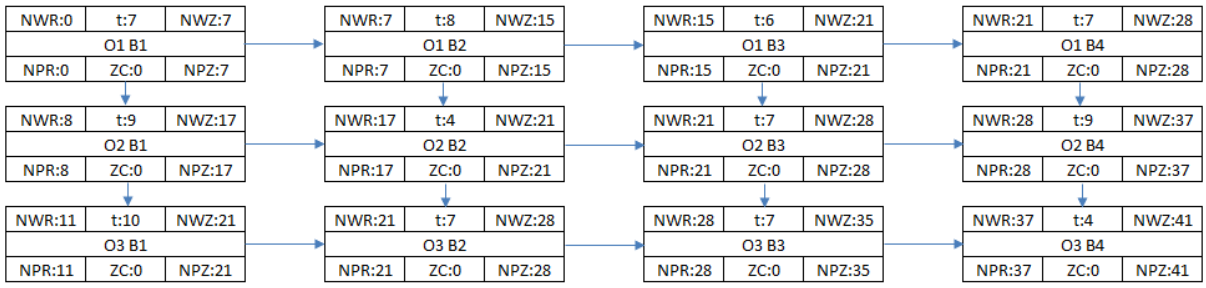
A.3	Ciągłość realizacji procesów na obiektach.	~45	45

Komentarz: Model, który pozwala wyznaczyć harmonogram zapewniający ciągłość pracy na obiektach roboczych. Czas realizacji takiego przedsięwzięcia jest dłuższy w stosunku do pierwszego przykładu o 1 dzień. Dodatkowo występują przerwy w pracy brygad. Podobne rozwiązanie uzyskamy korzystając z teorii sprzężeń czasowych. Wartość funkcji celu wyznaczono w oparciu o czas trwania, jednostkowe koszty pośrednie oraz współczynnik pozwalający wyznaczyć terminy najwcześniejsze i najpóźniejsze (z bardzo małym współczynnikiem). Wszystkie procesy są krytyczne.



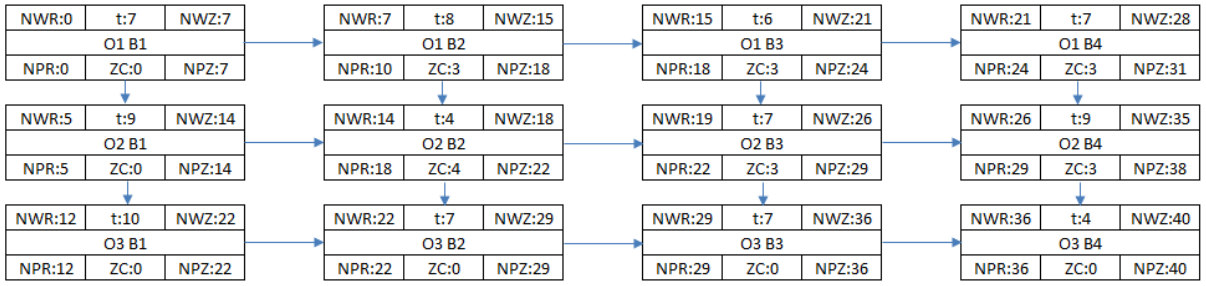
A.4	Ciągłość realizacji procesów na obiekcie drugim oraz ciągłość realizacji procesów dla brygady trzeciej.	60041	41

Komentarz: Model, który pozwala wyznaczyć harmonogram zapewniający ciągłość realizacji procesów na obiekcie drugim oraz ciągłość pracy brygady trzeciej. Paradoksalnie czas realizacji takiego przedsięwzięcia jest krótszy w stosunku do pierwszego przykładu o 3 dni. Wynika to z faktu, że ograniczenie na ciągłość pracy brygad i pracy na obiektach ma większą wartość w modelu optymalizacyjnym niż ograniczenia wynikające z metody CMP. Widać to dokładnie dla zadania O3B1, którego termin rozpoczęcia jest o 6 dni mniejszy niż termin zakończenia zadania O2B1. Problem ten można rozwiązać, korzystając z opracowanych zestawów wag – typ efektu A.8b, jednak wtedy należy się liczyć z brakiem możliwości dochowania ograniczeń ciągłości. Wartość funkcji celu uwzględnia zarówno koszty pośrednie, jak i kary związane z niedotrzymaniem ograniczeń w metodzie CPM (wspomniane zadanie O3B1). Wszystkie procesy są krytyczne.



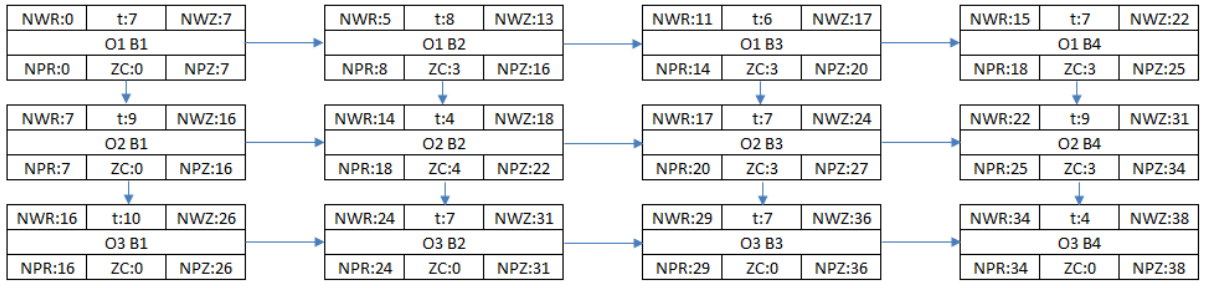
A.5	Możliwość wykonywania procesów na różnych obiektach przez jedną brygadę, z 2-dniowym nakładaniem się procesów.	~39.99	40

Komentarz: Możliwość wykonywania procesów na różnych obiektach przez jedną brygadę z 2-dniowym nakładaniem się realizacji procesów powoduje skrócenia czasu realizacji o 4 dni w stosunku do pierwszego przykładu. Ograniczenia w tym modelu są identyczne jak w modelu CPM, przy uwzględnieniu ujemnej zwłoki pomiędzy procesami realizowanymi przez tę samą brygadę. Wartość funkcji celu wyznaczono w oparciu o czas trwania, jednostkowe koszty pośrednie oraz współczynnik pozwalający wyznaczyć terminy najwcześniejsze i najpóźniejsze (z bardzo małym współczynnikiem). Istnieje jedna ścieżka krytyczna.



A.6	Możliwość wykonywania procesów na jednym obiekcie przez kilka brygad, z 2-dniowym nakładaniem się procesów.	~37.99	38

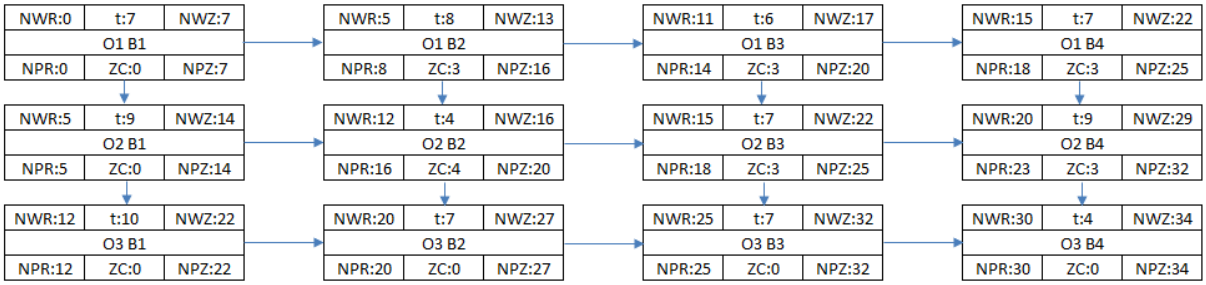
Komentarz: Możliwość wykonywania procesów na jednym obiekcie przez kilka brygad z 2-dniowym nakładaniem się procesów powoduje skrócenie czasu realizacji o 6 dni w stosunku do pierwszego przykładu. Ograniczenia w tym modelu są identyczne jak w modelu CPM, przy uwzględnieniu ujemnej zwłoki pomiędzy procesami realizowanymi na tym samym obiekcie. Wartość funkcji celu wyznaczono w oparciu o czas trwania, jednostkowe koszty pośrednie oraz współczynnik pozwalający wyznaczyć terminy najwcześniejsze i najpóźniejsze (z bardzo małym współczynnikiem). Istnieje jedna ścieżka krytyczna.



A.7	Kombinacja modelu A.5 i A.6.	~33.99	34

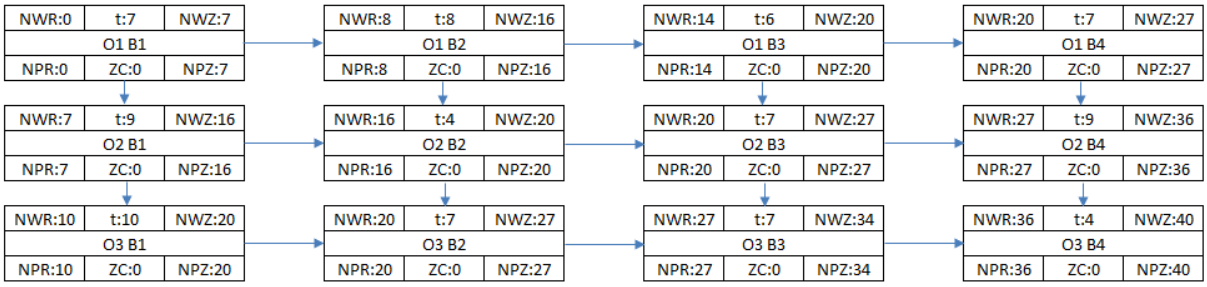
Komentarz: Możliwość wykonywania procesów na jednym obiekcie przez kilka brygad oraz wykonywania procesów na różnych obiektach przez jedną brygadę z 2-dniowym nakładaniem się procesów powoduje skrócenie czasu realizacji o 10 dni w stosunku do pierwszego przykładu. Ograniczenia w tym modelu są identyczne jak w

modelu CPM, przy uwzględnieniu ujemnej zwłoki pomiędzy procesami. Wartość funkcji celu wyznaczono w oparciu o czas trwania, jednostkowe koszty pośrednie oraz współczynnik pozwalający wyznaczyć terminy najwcześniejsze i najpóźniejsze (z bardzo małym współczynnikiem). Istnieje jedna ścieżka krytyczna.



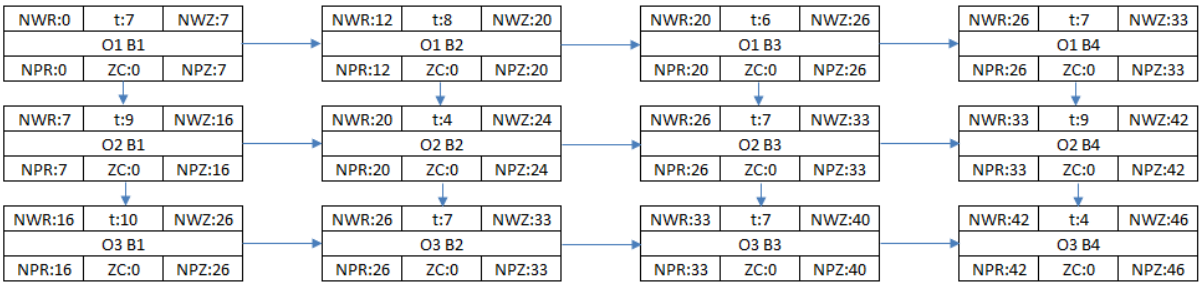
A.8a	Harmonogramowania priorytetowe z możliwością zachodzenia na siebie procesów: 1) ciągłość realizacji procesów przez brygadę B3; 2) ciągłość realizacji procesów na obiekcie O2; 3) ciągłość realizacji procesów przez brygadę B2.	80040	40

Komentarz: Przy narzuconych warunkach ciągłości uzyskano termin o 4 dni krótszy niż w pierwszym przykładzie. Zgodnie z preferencjami zostały spełnione wszystkie warunki ciągłości. Wymagało to jednak nałożenia się pracy brygady B1 na obiekcie O2 i O3 aż przez 6 dni oraz nałożenia się pracy brygad B2 i B3 na obiekcie O1 przez 2 dni. Niedotrzymanie ograniczeń metody CPM skutkowało naliczeniem kary w funkcji celu. Wszystkie procesy są krytyczne.



A.8b	Harmonogramowania priorytetowe bez możliwości zachodzenia na siebie procesów: 1) ciągłość realizacji procesów przez brygadę B3; 2) ciągłość realizacji procesów na obiekcie O2; 3) ciągłość realizacji procesów przez brygadę B2.	602000046	46

Komentarz: Przy narzuconych warunkach ciągłości oraz braku możliwości zachodzenia na siebie procesów uzyskano czas o 2 dni dłuższy, niż w przypadku pierwszego przykładu. Nie udało się spełnić wszystkich oczekiwań decydenta. Priorytetowa preferencja, a więc dotrzymanie ciągłości pracy brygady B3, została dotrzymana. Nie udało się dotrzymać ciągłości realizacji procesów na obiekcie O2 (nieciągłość wyniosła łącznie 6 dni) oraz nie dotrzymano ciągłości pracy dla brygady B2 (nieciągłość wyniosła 2 dni). Wartości funkcji celu określa niedotrzymane ograniczeń ciągłości (6 dni ze współczynnikiem 10^8 oraz 2 dni ze współczynnikiem 10^6). Wszystkie procesy są krytyczne.

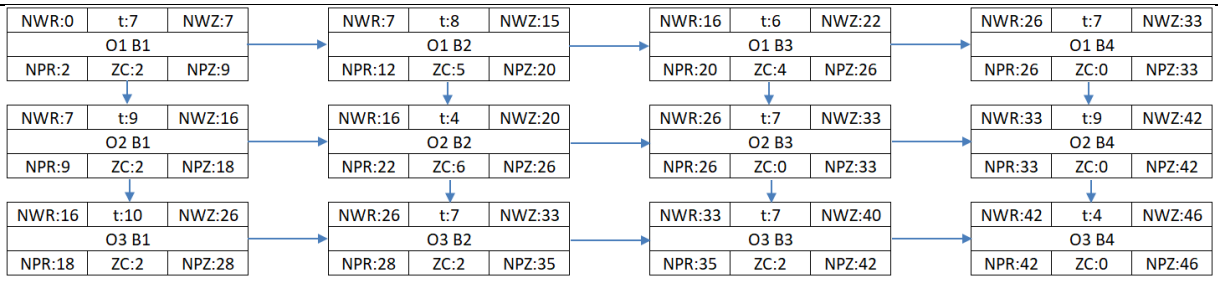


W tabeli 15 przedstawiono przykład obliczeniowy dla modelu zaprezentowanego w rozdziale 3.4.2. Wyniki uzyskano również dla danych z tabeli 13. Wyniki zostały otrzymane na podstawie kodu będącego załącznikiem 4 do niniejszej rozprawy.

Tabela 15: Otrzymane wyniki dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od B.1 do B.5

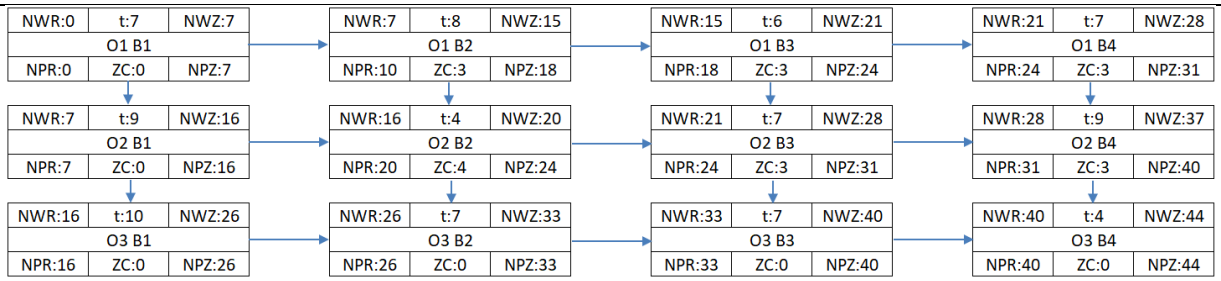
Typ efektu	Ograniczenia	Wartość funkcji celu	Czas trwania
B.1	Ograniczenia jak w metodzie CPM z uwzględnieniem sprzężeń diagonalnych dla terminów najwcześniejszych.	~45.99	46

Komentarz: Model realizujący ograniczenia jak w modelu CPM oraz dodatkowo uwzględniający sprzężenia diagonalne dla terminów najwcześniejszych rozpoczęcia. Najwcześniejszy termin rozpoczęcia procesów powinien być maksymalną wartością z terminu zakończenia poprzedników oraz z terminu rozpoczęcia roboty wykonywanej przez poprzednią brygadę na kolejnym obiekcie. W porównaniu do modelu A1 został wykorzystany zapas czasu niektórych procesów, aby dotrzymać ograniczeń (np. O1B3 lub O2B4), dodatkowo czas przedsięwzięcia został wydłużony o 2 dni. Ścieżka krytyczna przebiega już w sposób nieintuicyjny – bezpośredni poprzednicy procesu O2B3 posiadają zapas czasu całkowitego, natomiast ta praca nie może zostać wykonana później, bo proces O1B4 zostałby zrealizowany później, co spowoduje wydłużenie całego przedsięwzięcia.



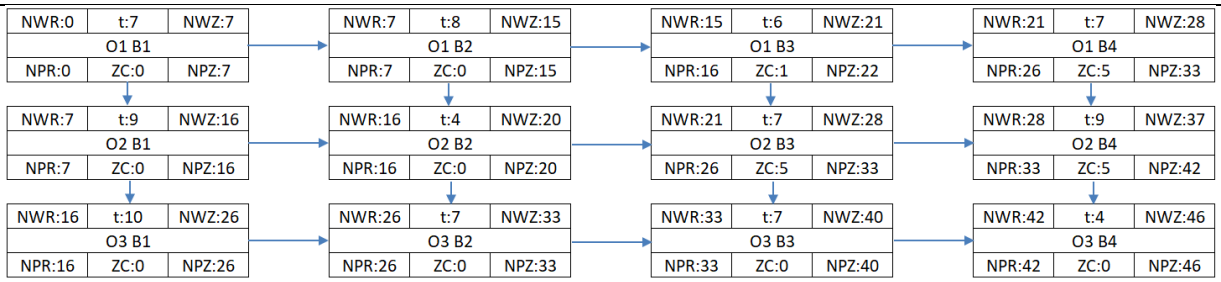
B.2	Ograniczenia jak w metodzie CPM z uwzględnieniem sprzężeń odwrotnych diagonalnych dla terminów najwcześniejszych rozpoczęcia.	~43.99	44

Komentarz: Model realizujący ograniczenia jak w modelu CPM oraz dodatkowo uwzględniający sprzężenia odwrotne diagonalne dla terminów najwcześniejszych rozpoczęcia. Najwcześniejszy termin rozpoczęcia procesu powinien być maksymalną wartością z terminu zakończenia poprzedników oraz z terminu rozpoczęcia procesu wykonywanego przez kolejną brygadę na poprzednim obiekcie. Wszystkie ograniczenia są spełnione, jednak to rozwiązanie nie różni się od rozwiązania uzyskanego przy typie efektu A.1.



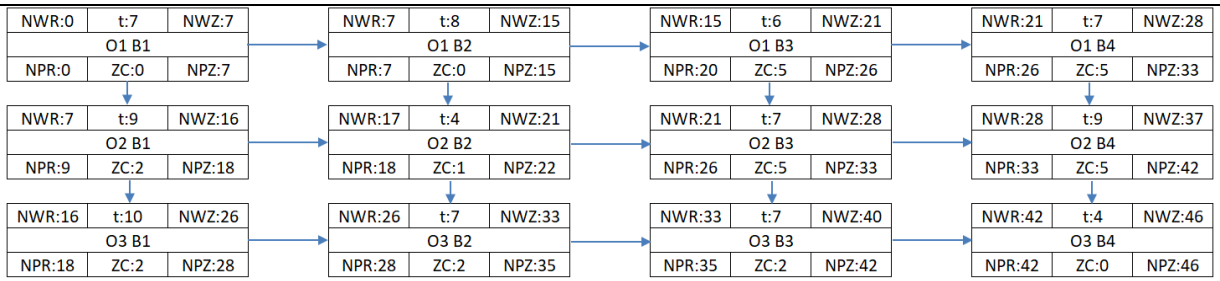
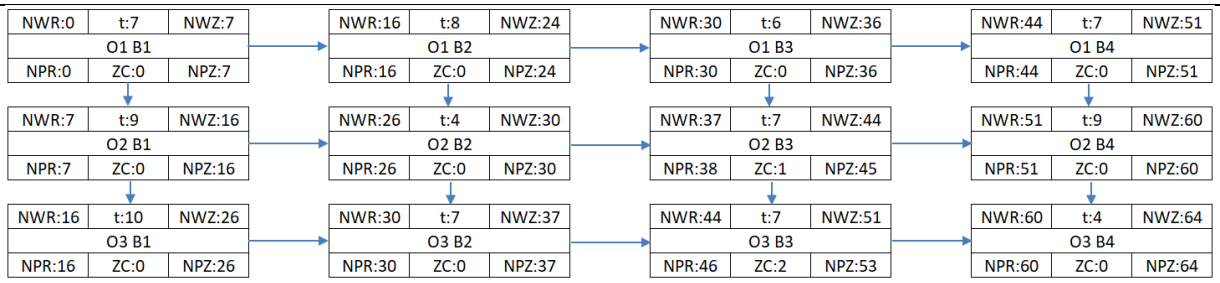
B.3	Ograniczenia jak w metodzie CPM z uwzględnieniem zerowych sprzężeń diagonalnych dla najpóźniejszych terminów rozpoczęcia.	~46	46

Komentarz: Model uwzględniający ograniczenia metody CPM oraz zerowe sprzężenia diagonalne dla najpóźniejszych terminów rozpoczęcia. Najpóźniejszych termin rozpoczęcia procesu powinien być równy najpóźniejszemu terminowi rozpoczęcia procesu realizowanego przez poprzednią brygadę na kolejnym obiekcie. Uzyskane rozwiązanie spełnia narzucone ograniczenia i daje dość nieoczywisty wynik. Proces O3B4 mógłby posiadać zapas czasu (procesy poprzedzające mają terminy najwcześniejszego zakończenia 40 i 37). Jednak ze względu na założenia metody CPM (najwcześniejszy termin zakończenia ostatniego procesu jest równy najpóźniejszemu terminowi zakończenia ostatniego procesu) zapas czasu ostatniego procesu musi wynosić 0.



B.4	Ograniczenia jak w metodzie CPM z narastającym o 1 dzień zapasem czasu dla procesu realizowanych przez k-tą brygadę.	~45.99	46

Komentarz: Model, który realizuje dość nietypowe założenia decyzyjne. Poza ograniczeniami wynikającymi z metody CPM, dodatkowym ograniczeniem jest wzrastający zapas czasu dla procesów realizowanych przez konkretną brygadę, w tym przypadku brygadę B2. Można zauważyć, że zakładane ograniczenia są spełnione. Brygada B2 posiada narastający o 1 dzień zapas całkowity czasu. Narastający zapas został zrealizowany nieintuicyjnie, ponieważ proces O2B2 mógłby mieć termin najwcześniejszy rozpoczęcia 16, jednak wtedy nie

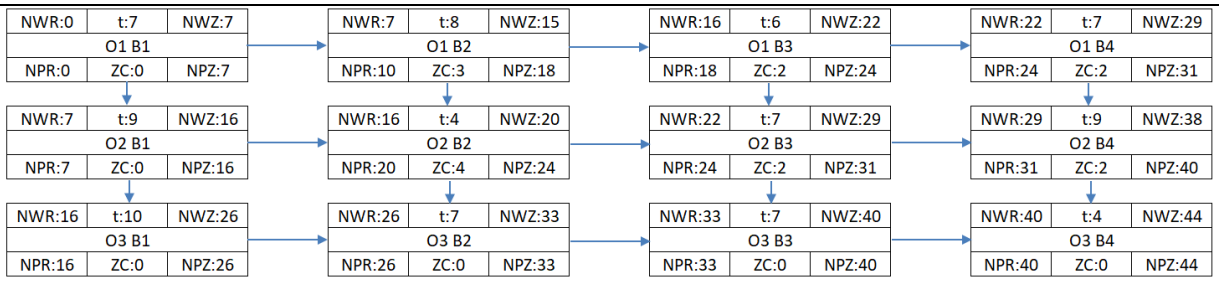
został by zachowany warunek na narastanie zapasu czasu. Z punktu widzenia logiki działania modelu nie jest to błąd.			
			
B.5	Ograniczenia jak w metodzie CPM z uwzględnieniem zerowych sprzężeń diagonalnych łączących najwcześniejsze terminy rozpoczęcia i zakończenia procesów.	~63.99	64
Komentarz: W tym modelu uwzględniono zerowe sprzężenia diagonalne pomiędzy najwcześniejszymi terminami rozpoczęcia i zakończenia procesów. Ograniczenie to oznacza, że termin najwcześniejszego rozpoczęcia jest równy terminowi najwcześniejszemu zakończenia procesu wykonywanego na kolejnym obiekcie przez poprzednią brygadę. Narzucone ograniczenia sprawiają, że czas realizacji przedsięwzięcia wynosi aż 64 dni. Wszystkie ograniczenia zostały spełnione. Niektóre procesy (O2B3 oraz O3B3) posiadają zapas czasu całkowitego.			
			

W tabeli 16 przedstawiono przykład obliczeniowy dla modelu zaprezentowanego w rozdziale 3.4.3. Wyniki uzyskano również dla danych z tabeli 13. Wyniki zostały otrzymane na podstawie kodu będącego załącznikiem 6 do niniejszej rozprawy.

Tabela 16: Otrzymane wyniki dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od C.1 do C.3

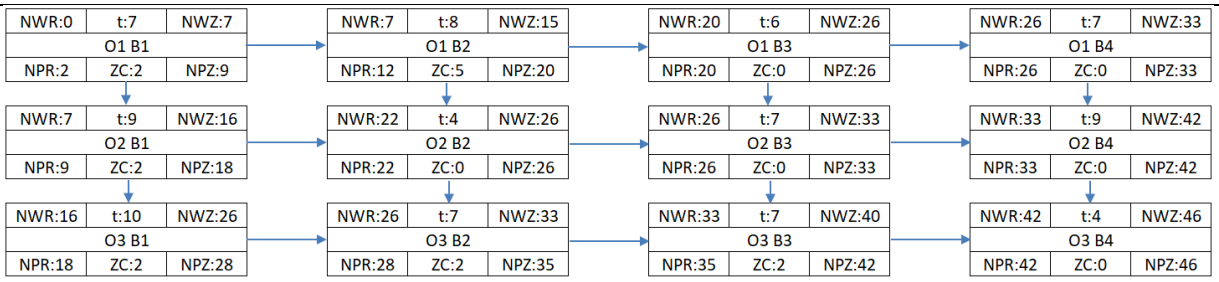
Typ efektu	**Ograniczenia**	**Wartość funkcji celu**	**Czas trwania**
C.1	Ograniczenia jak w metodzie CPM oraz zerowe sprzężenie diagonalne dla najwcześniejszych terminów rozpoczęcia pomiędzy procesami O3B1, O2B2, O1B3.	~44	44

Komentarz: Model, który spełnia ograniczenia metody CPM oraz zerowe sprzężenia diagonalne dla najwcześniejszych terminów rozpoczęcia dla procesów O3B1, O2B2 oraz O1B3. Rozwiązanie jest podobne do tego uzyskanego dla typu efektu A.1, jednak aby dochować zerowe sprzężenia diagonalne musiał zostać wykorzystany zapas całkowity procesu O1B3. Istnieje jedna ścieżka krytyczna i termin realizacji wynosi 44 dni.



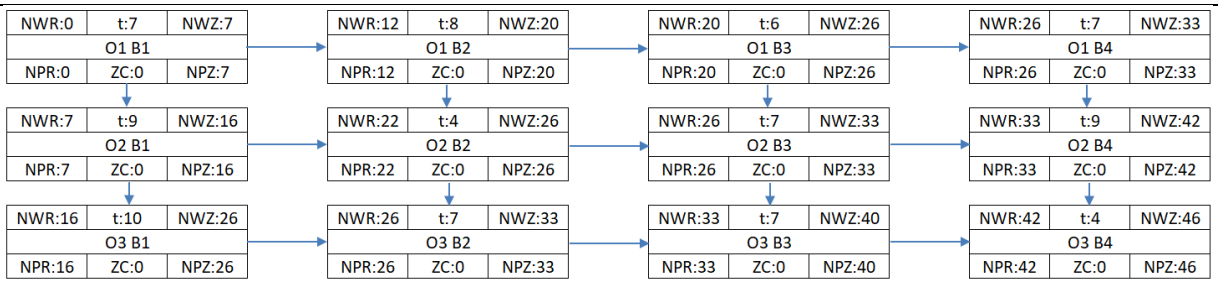
C.2	Ograniczenia jak w metodzie CPM oraz zerowe sprzężenie diagonalne dla najwcześniejszych terminów zakończenia pomiędzy robotami O3B1, O2B2, O1B3.	~46	46

Komentarz: Model, który spełnia ograniczenia metody CPM oraz zerowe sprzężenia diagonalne dla najwcześniejszych terminów zakończenia dla procesów O3B1, O2B2 oraz O1B3. Procesów O1B3 oraz O2B2 wykorzystały zapas czasu, aby dotrzymać ograniczenia związane z zerowym sprzężeniem diagonalnym w porównaniu do typu efektu A.1. Ścieżka krytyczna przebiega w sposób nieintuicyjny – pierwszy proces O1B1 posiada zapas czasu. Całkowity czas realizacji wynosi 46 dni.



C.3	Harmonogramowanie priorytetowe. Ograniczenia dochowane w podanej kolejności: 1) ograniczenia metody CPM; 2) zerowe sprzężenia diagonalne pomiędzy najwcześniejszymi terminami zakończenia pomiędzy procesami O3B1, O2B2, O1B3; 3) ciągłość realizacji procesów przez brygadę B3; 4) ciągłość realizacji procesów na obiekcie O2; 5) ciągłość realizacji procesów przez brygadę B2.	602000046.0	46

Komentarz: Model prezentujący zastosowanie koncepcji harmonogramowania priorytetowego i uwzględniający różne założenia decyzyjne. Model uwzględnia 5 typów ograniczeń. Nie wszystkie założenia decyzyjne udało się zrealizować. Nie udało się zachować ciągłości pracy na obiekcie O2 na poziomie 6 dni oraz nie udało się dochować ciągłości pracy brygady B2 na poziomie 2 dni. Poziom niedotrzymania założeń decyzyjnych ma odzwierciedlenie w funkcji celu która wyniosła: 602000046. Czas realizacji przedsięwzięcia wyniósł 46 dni.



4.3. Wybór optymalnych parametrów oraz przykład zastosowania modułu MCTS

Opracowana metoda optymalizacji dyskretnej oparta o metodę MCTS została przetestowana w kilku wariantach w celu wyboru najbardziej efektywnych parametrów. Zbadano wpływ różnych wzorów współczynnika, który służył do wyboru w procesie selekcji, na osiągane średnie wyniki. Wzory zostały opracowane przez autora w oparciu o wzór UCT występujący w literaturze (Kocsis & Szepesvári, 2006), (Gelly & Silver, 2007). Opracowane przez autora wzory będą oznaczane przez V_i:

$$V_1 = -\frac{\sum w_i}{\sum ss_i} \cdot \frac{1}{BV} + C \cdot \sqrt{\frac{\ln Pss_i}{s_i}}$$

$$V_2 = -\frac{\sum w_i}{\sum ss_i} \cdot \frac{1}{BV} + C \cdot \sqrt{\frac{\ln Bss}{s_i}}$$

$$V_3 = -\frac{\sum w_i}{\sum ss_i} \cdot \frac{1}{BV} + C \cdot \sqrt{\frac{\ln Rss}{s_i}}$$

$$V_4 = -\frac{BV_i}{BV} + C \cdot \sqrt{\frac{\ln Pss_i}{s_i}}$$

$$V_5 = -\frac{BV_i}{BV} + C \cdot \sqrt{\frac{\ln Bss}{s_i}}$$

$$V_6 = -\frac{BV_i}{BV} + C \cdot \sqrt{\frac{\ln Rss}{s_i}}$$

gdzie:

s_i – liczba wszystkich symulacji dla węzła i;

Pss_i – suma liczby symulacji wszystkich następników dla bezpośredniego poprzednika węzła i;

Bss – suma liczby symulacji dla najlepszego węzła poniżej węzła stanowiącego korzeń (root) drzewa;

Rss – suma liczby symulacji dla węzła stanowiącego korzeń (root) drzewa;

$\sum w_i$ – suma wszystkich uzyskanych wartości funkcji celu dla węzła i;

$\sum ss_i$ – liczba symulacji zakończonych sukcesem dla węzła i;

BV – najlepsza odnaleziona wartość funkcji celu;

BV_i – najlepsza odnaleziona wartość funkcji celu dla węzła i.

Dokonano również symulacji wpływu parametru C na średni uzyskiwany wynik. Wartość współczynnika C jest ustalana empirycznie. W zależności od rozwiązywanego problemu, parametr C przyjmuje różne wartości. Dla problemu jednorękiego bandyty wartość ta wynosi 1,41 (Kocsis & Szepesvári, 2006). Do obliczeń przyjęto wartości parametru C ze zbioru $\{0{,}001; 0{,}005; 0{,}01; 0{,}05; 0{,}1; 0{,}5; 1; 1{,}41; 2; 5; 10; 20; 50\}$.

Zbadano również wpływ wyboru najlepszego węzła. Zaproponowano trzy różne sposoby, oznaczane przez S_i:

S_1 - największa wartość zmiennej V_i

S_2 - najmniejsza wartość funkcji celu

S_3 - największą liczba przeprowadzonych symulacji

Aby wybrać wstępne parametry metody optymalizacyjnej MCTS przeprowadzono 20 symulacji dla każdej konfiguracji parametrów. Łączna liczba wszystkich kombinacji wyniosła 234 (13 wartości parametru C, 6 różnych sposobów wyznaczenia współczynnika V_i, 3 sposoby wyboru węzła S_i). Na każdą symulację przeznaczono około 60 sekund. Analizowano znany w literaturze przykład oznaczany ta001 (Taillard, 1993), składający się z 20 obiektów, realizowanych przez 5 brygad. Funkcją celu jest C_{max}, czyli czas trwania przedsięwzięcia. Najlepszym znanym rozwiązaniem jest 1278.

W załączniku 7 (w postaci tabeli) oraz w załączniku 8 (w postaci wykresu) przedstawiono średnią wartość funkcji celu w zależności od parametru C dla wszystkich 18 kombinacji V_i oraz S_i. Po analizie danych okazało się, że uzyskiwane rozwiązania z zastosowaniem wzorów V_4,V_5 i V_6 (niezależnie od sposobu wyznaczania S_i) znacząco odstają od pozostałych zestawów parametrów. Następnie odrzucono sposób wyboru węzła S_1, ponieważ wyniki znacząco różniły

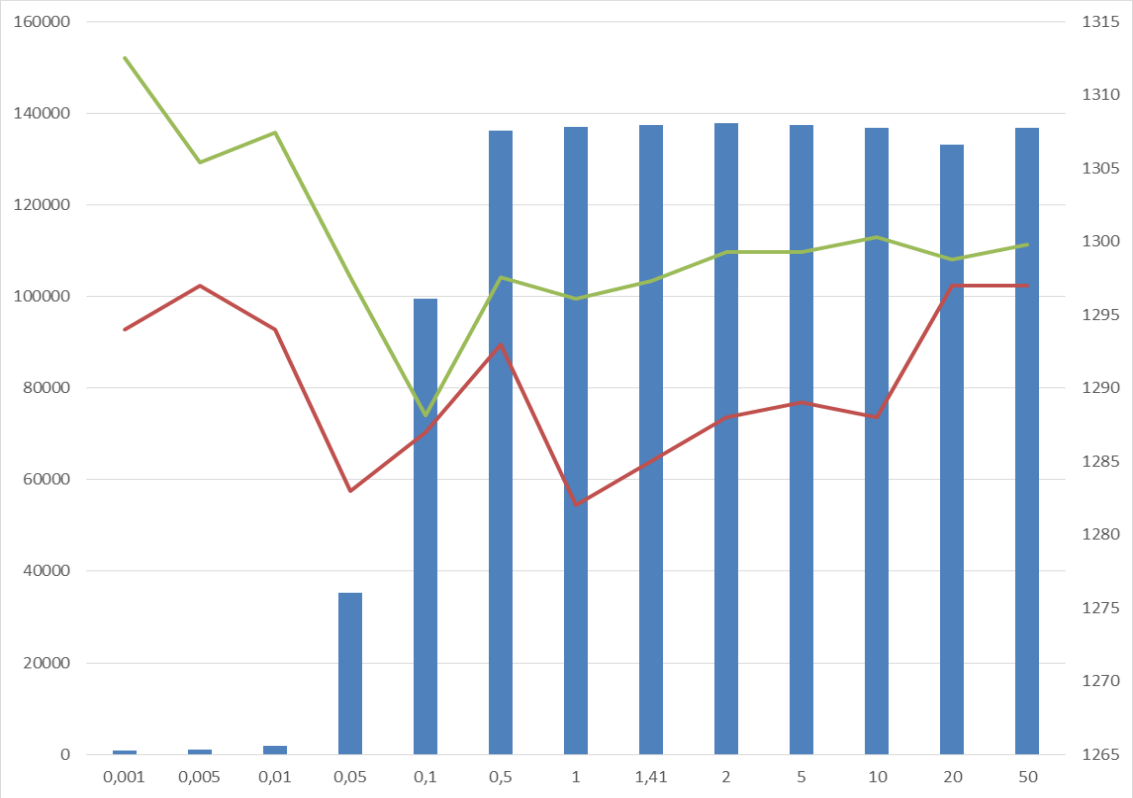
się od wyników uzyskiwanych dla pozostałych zestawów parametrów - zwłaszcza dla dużych wartości parametru C. Pod uwagę zostały wzięte 4 zestawy parametrów: (S_2, V_2), (S_2, V_3), (S_3, V_2), (S_3, V_3). Ostatecznie najmniejsze średnie wartości uzyskano dla wzoru V_3 oraz wyboru węzła S_2. W dalszych analizach wzięty pod uwagę został tylko ten zestaw.

Dodatkowo na rysunku 25 została przedstawiona zależność średniej wartości funkcji celu (zielony wykres – oś pomocnicza), najlepszej osiągniętej wartości funkcji celu (wykres czerwony – oś pomocnicza) oraz średniej liczby symulacji (wykres słupkowy – oś główna), w zależności od przyjętej wartości parametru C dla wzoru V_3 i sposobu wyboru węzła S_2.

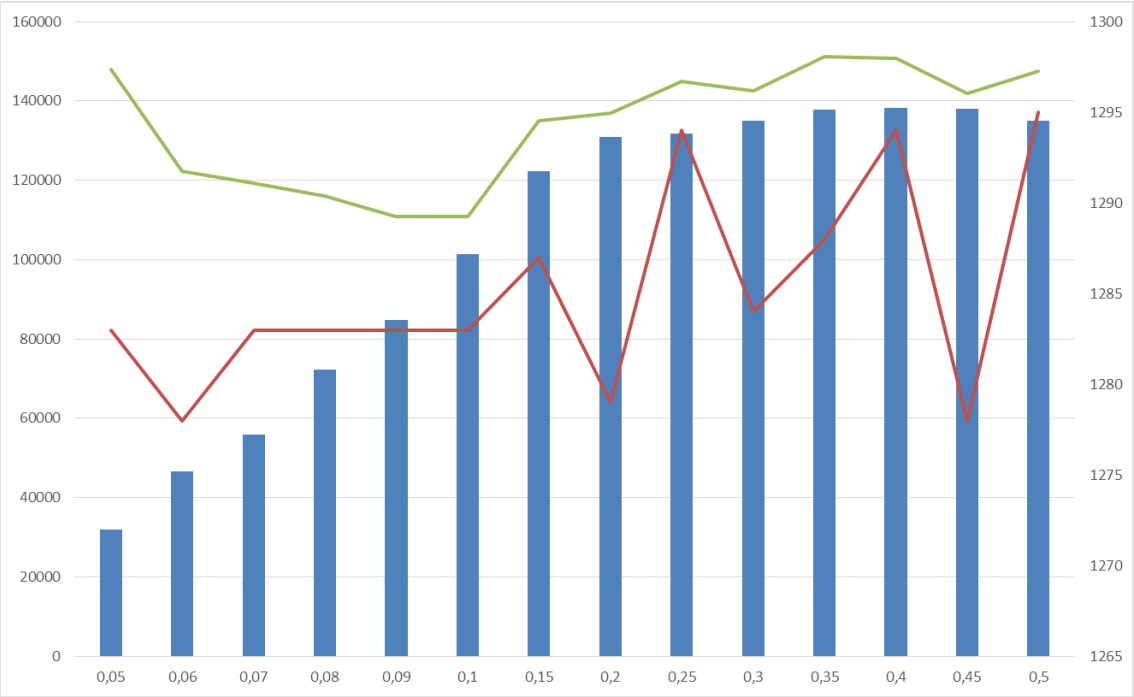
Dla małych wartości parametru C (od 0,001 do 0,01) średnia liczba symulacji jest bardzo mała. Oznacza to, że w czasie wykonywania algorytmu w procedurze wyboru dochodziło bardzo często do eksploatacji, czyli rozwijania drzewa w kierunku najlepszego znalezionego rozwiązania. Jest to niekorzystne zjawisko, ponieważ nie pozwala ono na ucieczkę algorytmu z minimów lokalnych. W związku z tym, średnia wartość funkcji celu dla małych wartości C jest stosunkowo duża (ponad 1310). Wraz ze zwiększającą się wartością parametru C średnia wartość funkcji celu maleje. Najlepsze średnie rozwiązanie uzyskano dla wartości C=0,1, jednak wtedy liczba symulacji wynosi zaledwie ok. 100000, podczas gdy dla wartości C=0,2 algorytm wykonał o 40% więcej symulacji. Dla zwiększającej się wartości parametru C powyżej 0,2 można zaobserwować rosnącą wartość funkcji celu. Oznacza to, że w czasie wykonywania algorytmu dochodzi znacznie częściej do eksploracji i metoda MCTS zaczyna przypominać klasyczną metodę Monte Carlo. Jest to również niekorzystne zjawisko, ponieważ nie wykorzystuje się wtedy możliwości eksploatacji najbardziej obiecujących fragmentów drzewa. Idealną sytuacją jest moment, w którym dochodzi do równowagi pomiędzy eksploatacją a eksploracją. Na podstawie przeprowadzonych obliczeń przyjęto wstępnie, że wartość współczynnika C należy ustalać na poziomie od 0,05 do 0,5.

W celu wyznaczenia dokładniejszej wartości parametru C dokonano szczegółowej analizy dla wartości z przedziału $\{0{,}05; 0{,}06; 0{,}07; 0{,}08; 0{,}09; 0{,}10{,}15; 0{,}2; 0{,}25; 0{,}3; 0{,}35; 0{,}4; 0{,}45; 0{,}5\}$ dla (S_2, V_3). Dla każdej wartości parametru C wykonano 20 iteracji algorytmu. Na rysunku 26 została przedstawiona zależność średniej wartości funkcji celu (zielony wykres – oś pomocnicza), najlepszej osiągniętej wartości funkcji celu (wykres czerwony – oś pomocnicza) oraz średniej liczby symulacji (wykres słupkowy – oś główna) w zależności od przyjętej wartości parametru C dla wzoru V_3 i sposobu wyboru węzła S_2. Można zaobserwować, że najlepszą średnią wartość funkcji celu osiągnięto dla wartości parametru C na poziomie $0{,}09$ oraz $0{,}1$. W pozostałych przypadkach osiągano odbiegające wartości funkcji celu. Dodatkowo na wykresie słupkowym widać, że osiągnięto balans pomiędzy eksploracją i eksploatacją. W kolejnych przykładach wykorzystujących zmodyfikowaną metodę MCTS przyjęto wartość parametru C jako $0{,}1$.

W załączniku 9 przedstawiono przebieg 20 różnych iteracji metody MCTS dla (S_2, V_3) oraz wartości parametru $C = 0{,}1$. W każdym wierszu jest przedstawiony odnaleziony najlepszy wynik dla danego przebiegu algorytmu. W pierwszej kolumnie przedstawiono numer iteracji, w drugiej czas działania programu, w trzeciej wartość funkcji celu, w czwartej węzeł, na którym znaleziono aktualne najlepsze rozwiązanie, w piątej, podczas której iteracji w pętli wewnętrznej metody MCTS odnaleziono aktualne najlepsze rozwiązanie, w szóstej najlepsze aktualnie uszeregowanie. Można zauważyć, że podczas każdego przebiegu algorytmu wartość funkcji celu maleje, co wskazuje na skuteczność działania metody. Znaczna część rozwiązań suboptymalnych jest odnajdywana w węźle początkowym drzewa metody MCTS.



Rysunek 25: Zależność średniej wartości funkcji celu (zielony wykres – oś pomocnicza), najlepszej osiągniętej wartości funkcji celu (wykres czerwony – oś pomocnicza) oraz średniej liczby symulacji (wykres słupkowy – oś główna) w zależności od przyjętej wartości parametru C dla wzoru V_3 i sposobu wyboru węzła S_2 – C z przedziału $\{0{,}001; 0{,}005; 0{,}01; 0{,}05; 0{,}1; 0{,}5; 1; 1{,}41; 2; 5; 10; 20; 50\}$



Rysunek 26: Zależność średniej wartości funkcji celu (zielony wykres – oś pomocnicza), najlepszej osiągniętej wartości funkcji celu (wykres czerwony – oś pomocnicza) oraz średniej liczby symulacji (wykres słupkowy – oś główna) w zależności od przyjętej wartości parametru C dla wzoru V_3 i sposobu wyboru węzła S_2 – C z przedziału $\{0{,}05; 0{,}06; 0{,}07; 0{,}08; 0{,}09; 0{,}1; 0{,}15; 0{,}2; 0{,}25; 0{,}3; 0{,}35; 0{,}4; 0{,}45; 0{,}5\}$

W celu zaprezentowania skuteczności działania metody MTCS w zagadnieniu szeregowania zadań, postanowiono przetestować opracowaną metodę na ogólnodostępnych przykładach (Taillard, 1993). Wybrano 7 przykładów o różnych wielkościach, co zostało przedstawione w tabeli. Każdy przykład został przeliczony proponowaną metodą MTCS dziesięć razy. Zaprezentowano najlepsze uzyskane rozwiązanie, średnią wartość rozwiązania oraz błąd bezwzględny uzyskanej wartości średniej. Należy zwrócić uwagę na czas wykonania kodu programu, który oscyluje od 5 minut do 35 minut, w zależności od wielkości zadania (dla jednej iteracji). Najlepsze znane rozwiązania zostały uzyskiwane przez wielodniowe obliczenia na superkomputerach. Prezentowane wyniki zostały uzyskane przez wykonywane obliczeń na komputerze osobistym średniej jakości. Nie należy się więc dziwić, że procentowa skuteczność metody nie jest zbyt dobra, jednak szczegółowa analiza przebiegu obliczeń (załącznik 10 – dla zadania ta092) pozwala wykazać, że metoda dąży do coraz lepszych rozwiązań. Dodatkowo należy zwrócić uwagę, że przedsięwzięcia budowlane zazwyczaj składają się z nie więcej niż 50 obiektów (najczęściej nie więcej niż 20). Analizę przeprowadzono jednak nawet dla 100 i 200 obiektów, uzyskując zadowalające (jak na zaangażowaną moc obliczeniową) wyniki, które zaprezentowano w tabeli 17.

Tabela 17 Analiza eksperymentalna

Nazwa przykładu	Liczba obiektów	Liczba brygad	Liczba możliwych rozwiązań	Najlepsze znane rozwiązanie [dni]	Czas wykonania kodu [sekundy]	Najlepsze uzyskane rozwiązanie [dni]	Średnia wartość z 10 iteracji [dni]	Błąd względny średniej wartości [%]
ta006	20	5	$2{,}43 \cdot 10^{23}$	1195	300	1199	1213,6	1,47
ta015	20	10	$2{,}43 \cdot 10^{23}$	1419	600	1501	1519,0	7,05
ta027	20	20	$2{,}43 \cdot 10^{23}$	2273	900	2332	2352,0	3,48
ta041	50	10	$3{,}04 \cdot 10^{64}$	2991	1200	3207	3251,4	8,71
ta062	100	5	$9{,}33 \cdot 10^{157}$	5268	1500	5344	5374,4	2,02
ta073	100	10	$9{,}33 \cdot 10^{157}$	5676	1800	5985	6063,4	6,83
ta092	200	10	$7{,}88 \cdot 10^{374}$	10480	2100	11234	11327,8	8,09
							Średni błąd	5,38

Zaproponowaną metodę MCTS porównano również z dwoma innymi metodami optymalizacji dyskretnej: Monte Carlo (MC) oraz Symulowane Wyżarzanie (SA). Każdy przykład został przeliczony poszczególną metodą 10 razy. Obie metody miały taki sam czas wykonania kodu (5 minut). W tabeli 18 przedstawiono uzyskane wyniki oraz porównanie uzyskanych rozwiązań dla wszystkich 3 metod wraz z najlepszym znanym rozwiązaniem. Jak widać, metoda MC osiąga najgorsze wyniki – średni błąd wynosi 7,34%. Druga jest metoda MCTS ze średnim błędem na poziomie 5,38%, co jest lepszym wynikiem niż MC, ale gorszym niż SA. Metoda SA uzyskała średni błąd na poziomie 0,94%, co jest najlepszym wynikiem ze wszystkich metod. Należy zwrócić uwagę, że metoda SA ma dużą skłonność do ucieczki z minimów lokalnych oraz jest cenioną metodą w optymalizacji kombinatorycznej. Zaproponowana w dysertacji metoda osiąga gorsze wyniki, natomiast jej największą zaletą jest dobre radzenie sobie z problemami, w których występuje dużo ograniczeń ze względu na kolejność wykonania obiektów. Ograniczenia te zostały opisane w rozdziale 3.3.4.

Tabela 18. Porównanie metod MC, SA i MCTS

Nazwa przykładu	Najlepsze znane rozwiązanie [dni]	Najlepsze rozwiązanie MC [dni]	Średnie rozwiązanie MC [dni]	Błąd względny średniej wartości MC [%]	Najlepsze rozwiązanie SA [dni]	Średnie rozwiązanie SA [dni]	Błąd względny średniej wartości SA [%]	Najlepsze rozwiązanie MCTS [dni]	Średnie rozwiązanie MCTS [dni]	Błąd względny średniej wartości MCTS [%]
ta006	1195	1226	1240	3,48	1195	1206,7	0,98	1199	1213,6	1,47
ta015	1419	1524	1543,2	8,46	1425	1434	1,06	1501	1519,0	7,05
ta027	2273	2387	2404,4	5,60	2276	2302,7	1,31	2332	2352,0	3,48
ta041	2991	3328	3390,5	12,89	3025	3035,9	1,50	3207	3251,4	8,71
ta062	5268	5400	5426,5	2,97	5280	5284,1	0,31	5344	5374,4	2,02
ta073	5676	6133	6159,4	8,54	5679	5698,5	0,40	5985	6063,4	6,83
ta092	10480	11404	11496,1	9,46	10524	10587,9	1,03	11234	11327,8	8,09
			Średni błąd	7,34		Średni błąd	0,94		Średni błąd	5,38

Zaproponowana metoda MCTS z ustalonymi wartościami parametrów oraz zaproponowanym sposobem eksploracji i eksploatacji okazała się skuteczniejsza niż metoda MC. Uzyskane wyniki są satysfakcjonujące na potrzeby niniejszej pracy. Należy zwrócić też uwagę, że celem pracy nie jest poszukiwanie optymalnej metody optymalizacji dyskretnej.

Dokonano jeszcze jednej analizy, tym razem z uwzględnieniem ograniczeń opisanych w rozdziale 3.3.4. Dla dwóch przykładowych zadań, tj. ta006 oraz ta041 została przeprowadzona optymalizacja z zastosowaniem opracowanej metody. Narzucono następujące ograniczenia na kolejność wykonania obiektów:

- „3=2” – obiekt numer 3 powinien być wykonany jako 2 w kolejności;
- „4>5” – obiekt numer 5 musi zostać wykonany bezpośrednio po obiekcie o numerze 4;
- „14>11” – obiekt numer 11 musi zostać wykonany bezpośrednio po obiekcie o numerze 14;
- „8>>9” – obiekt o numerze 9 musi zostać wykonany po obiekcie o numerze 8;
- „6<>7” – obiekt o numerze 6 i 7 muszą zostać wykonane bezpośrednio po sobie (6 po 7 albo 7 po 6);
- „18_19_1” – obiekty o numerach 18, 19 i 1 muszą zostać wykonane jeden po drugim, ale w dowolnej kolejności.

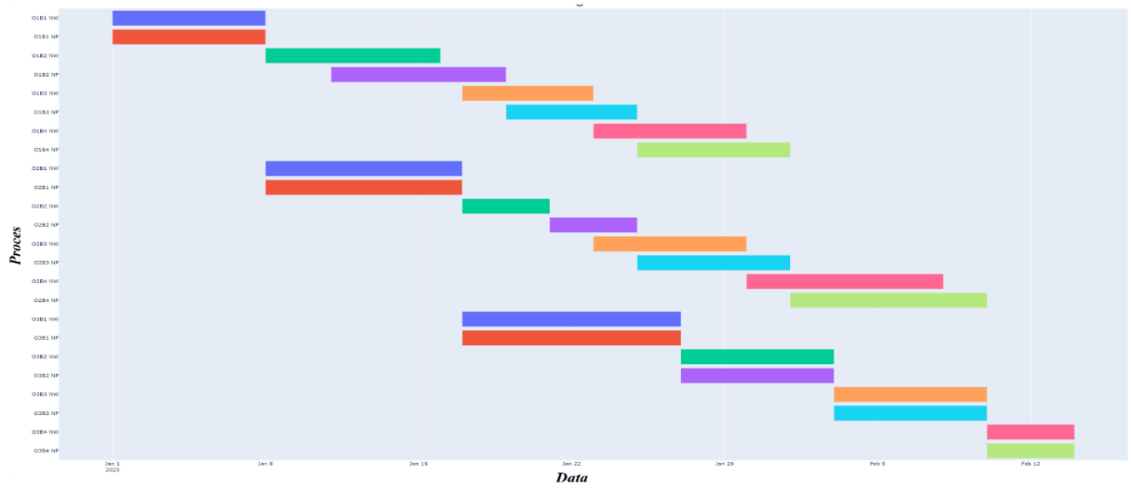
Oba przykłady zostały przeliczone 10 razy dla czasu wykonania odpowiednio 900 i 1800 sekund. Dla zadania ta006 uzyskano wynik średni 1344,3 oraz minimalny 1326. Kolejność dla najlepszego uzyskanego wyniku prezentuje się następująco: [16, 3, 13, 14, 11, 20, 17, 4, 5, 15, 10, 7, 6, 8, 12, 9, 2, 1, 18, 19]. Wszystkie narzucone ograniczenia na kolejność zostały dochowane. Dla zadania ta041 wynik średni wyniósł 3242,6 oraz minimalny 3170. Kolejność dla najlepszego uzyskanego rozwiązania to: [25, 3, 20, 19, 18, 1, 22, 33, 49, 43, 38, 7, 6, 44, 46, 21, 15, 36, 42, 10, 35, 37, 13, 8, 14, 11, 23, 28, 4, 5, 9, 47, 32, 31, 34, 40, 29, 2, 16, 50, 12, 45, 41, 26, 27, 17, 48, 30, 24, 39]. Wszystkie ograniczenia zostały dochowane. W załączniku 12 został przedstawiony przykładowy przebieg algorytmu dla zadania ta041. Należy zwrócić uwagę, że w toku obliczeń programu wartość funkcji celu maleje. Można po tym wnioskować, że algorytm zbiega do rozwiązania suboptymalnego. Dodatkowo każde uzyskiwane rozwiązanie jest rozwiązaniem dopuszczalnym ze względu na narzucone ograniczenia kolejności wykonania obiektów.

4.4. Przykład zastosowania modułu użytkowego

Moduł użytkowy został przetestowany na przykładzie terminów uzyskanych w rozdziale 4.2 dla modelu C1. Termin rozpoczęcia przedsięwzięcia został ustalony na 1 stycznia 2023 roku. Na rysunku 27 znajduje się widok tabel w programie Excel, zawierających nazwę procesu, termin najwcześniejszego rozpoczęcia i zakończenia oraz termin najpóźniejszego rozpoczęcia i zakończenia. Uzyskane dane można przekopiować do programu Project w celu dalszej obróbki harmonogramu. Dodatkowo na rysunku 28 został zaprezentowany roboczy wykres Gantta uzyskany dla tego samego przykładu. Każdy proces jest wyświetlany w dwóch wierszach (dla czasów najwcześniejszych i najpóźniejszych). Każda brygada ma swój własny kolor (różny dla terminów najpóźniejszych i najwcześniejszych). Aby zwiększyć użyteczność oraz komfort użytkownika, należałoby wyniki eksportować do programu do harmonogramowania (np. Project), jednak leży to poza zakresem tej rozprawy. Kod programu w języku Python pozwalający uzyskać prezentowane wyniki został zawarty w załączniku 13.

	A	B	C	D	E
1	**Nazwa**	**Najwcześniejsze rozpoczęcie**	**Najwcześniejsze zakończenie**	**Najpóźniejsze rozpoczęcie**	**Najpóźniejsze zakończenie**
2	O1B1	2023-01-01	2023-01-08	2023-01-01	2023-01-08
3	O1B2	2023-01-08	2023-01-16	2023-01-11	2023-01-19
4	O1B3	2023-01-17	2023-01-23	2023-01-19	2023-01-25
5	O1B4	2023-01-23	2023-01-30	2023-01-25	2023-02-01
6	O2B1	2023-01-08	2023-01-17	2023-01-08	2023-01-17
7	O2B2	2023-01-17	2023-01-21	2023-01-21	2023-01-25
8	O2B3	2023-01-23	2023-01-30	2023-01-25	2023-02-01
9	O2B4	2023-01-30	2023-02-08	2023-02-01	2023-02-10
10	O3B1	2023-01-17	2023-01-27	2023-01-17	2023-01-27
11	O3B2	2023-01-27	2023-02-03	2023-01-27	2023-02-03
12	O3B3	2023-02-03	2023-02-10	2023-02-03	2023-02-10
13	O3B4	2023-02-10	2023-02-14	2023-02-10	2023-02-14

Rysunek 27: Prezentacja możliwości modułu użytkowego – eksport danych do tabeli Excel. Terminy uzyskane dla zadania C1 i terminu rozpoczęcia 1.01.2023 r.



Rysunek 28 Prezentacja możliwości modułu użytkowego – wykres Gantta. Terminy uzyskane dla zadania C1 i terminu rozpoczęcia 1.01.2023 r.

4.5. Przykład kompleksowy

4.5.1. Przykład 1

Skuteczność opracowanego modelu została przetestowana na przykładzie uwzględniającym wszystkie opracowane moduły. Wygenerowano przykład losowy dla przedsięwzięcia budowy osiedla domów jednorodzinnych składającego się z 20 obiektów oraz realizowanego przez wyspecjalizowanych 5 brygad roboczych.

W tabeli 19 przedstawiono, w jakich zakresach były generowane dane. Koszty pośrednie przyjęto na poziomie 4,0 za każdy dzień realizacji przedsięwzięcia, koszty niedotrzymania terminu dyrektywnego na poziomie 3,0 za każdy dzień opóźnienia, nagrodę za wcześniejsze zrealizowanie obiektu na poziomie 0,2 za każdy dzień, koszt nieciągłości pracy brygad B1, B3, B4, B5 na poziomie 0, a brygady B2 na poziomie 0,2 za każdy dzień nieciągłości pracy. Nie ograniczano brygad i obiektów terminami dostępności. Wszystkie omawiane wartości kosztów podane są w tys. zł (zaokrąglane do 0,1), a wartości czasu w dniach (zaokrąglane do całości).

Dodatkowo zostało nałożone ograniczenie z wykorzystaniem harmonogramowania priorytetowego. Wartości parametrów zostały tak dobrane, aby:

- najważniejszy jest warunek na ciągłość pracy brygady B4;
- drugim w kolejności jest warunek na ciągłość pracy na obiekcie O19;
- następnie należy dochować ograniczeń metody CPM.

Dodatkowo zostało narzucone ograniczenie na kolejność realizacji obiektów:

- „1=1” – obiekt O1 musi być zrealizowany jako 1 w kolejności;
- „2>3” – obiekt O3 musi być zrealizowany bezpośrednio po obiekcie O2;
- „3>4” – obiekt O4 musi być zrealizowany bezpośrednio po obiekcie O3;
- „6>>7” – obiekt O7 musi być zrealizowany po obiekcie O6;
- „8<>9” – obiekty O8 i O9 muszą być zrealizowane jeden po drugim;
- „10_11_12” – obiekty O10, O11 i O12 muszą być zrealizowane jako jedna grupa (jeden po drugim, ale w dowolnej kolejności).

W modelu optymalizacji dyskretnej MCTS przyjęto wybrane wcześniej parametry: wzór V_3, wybór węzła metodą S_2, wartość parametru $C = 0{,}1$. Czas wykonania jednej iteracji ustawiono na 5 godzin. Optymalizację dyskretną zmodyfikowaną metodą MCTS powtórzono 5 razy. Wygenerowane dane wejściowe oraz wartości parametrów znajdują się w załączniku 14, który przedstawia kod programu rozwiązujący dany przykład.

Tabela 19: Zakresy danych do wygenerowania przykładu obliczeniowego

Zmienna:	kgr		tgr		t		a	
Rozkład:	Ciągły		Dyskretny		Dyskretny		Ciągły	
	min	max	min	max	min	max	min	max
B1	30	40	10	20	(1,1,1)	(3,3,3)	(-3,-3,-3)	(-1,-1,-1)
B2	60	80	40	50	(1,1,1)	(3,3,3)	(-3,-3,-3)	(-1,-1,-1)
B3	50	70	30	40	(1,1,1)	(3,3,3)	(-3,-3,-3)	(-1,-1,-1)
B4	60	70	30	50	(1,1,1)	(3,3,3)	(-3,-3,-3)	(-1,-1,-1)
B5	30	40	30	60	(1,1,1)	(3,3,3)	(-3,-3,-3)	(-1,-1,-1)

W tabeli 20 zostały zestawione uzyskane wyniki dla pięciu iteracji. W tabeli przedstawiono uzyskane uszeregowanie, wartość funkcji celu, koszty całkowite, czas realizacji oraz poziom niedotrzymania ograniczeń w metodzie harmonogramowania priorytetowego. Wartości w trzech ostatnich kolumnach wskazują o ile dni nie udało się dotrzymać ograniczeń. Wartość 0 oznacza, że ograniczenie zostało dotrzymane. Dodatkowo dokonano obliczeń dla uszeregowania bazowego (tzn. [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20]). W tabeli zostało ono oznaczone jako iteracja 0.

Tabela 20: Wyniki uzyskane dla pięciu iteracji (oraz uszeregowania początkowego – iteracja 0) wykonane dla przykładu 1.

Iteracja	Uszeregowanie	Funkcja celu	Koszty całkowite	Czas realizacji	Niedotrzymanie ciągłości B4	Niedotrzymanie ciągłości O19	Niespełnienie warunków CPM
0	[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]	409267,5	9267,5	1042	0	0	40
1	[1,19,11,10,12,9,8,18,20,13,15,4,5,17,14,16,6,7,2,3]	88699,9	8699,9	1032	0	0	8
2	[1,13,9,8,19,17,11,10,12,18,14,4,5,20,16,15,2,3,6,7]	28941,4	8941,4	1041	0	0	2
3	[1,19,11,12,10,17,15,6,4,5,7,20,9,8,14,13,16,18,2,3]	88720,9	8720,9	1031	0	0	8
4	[1,18,8,9,13,15,19,17,11,10,12,6,2,3,14,20,16,7,4,5]	28792,1	8792,1	1035	0	0	2
5	[1,2,3,6,10,11,12,15,14,7,16,18,17,20,13,4,5,9,8,19]	9035,6	9035,6	1012	0	0	0

Najmniejszą wartość funkcji celu uzyskano dla iteracji 5 i wyniosła ona 9035,6; najwyższą dla iteracji 3 o wartości 88720,9. Dla najlepszego rozwiązania udało się dotrzymać wszystkich ograniczeń technologiczno-organizacyjnych, w pozostałych rozwiązaniach niedotrzymanie ograniczeń wynosiło od 2 do 8 dni. Dla iteracji 5 osiągnięto również najkrótszy czas realizacji

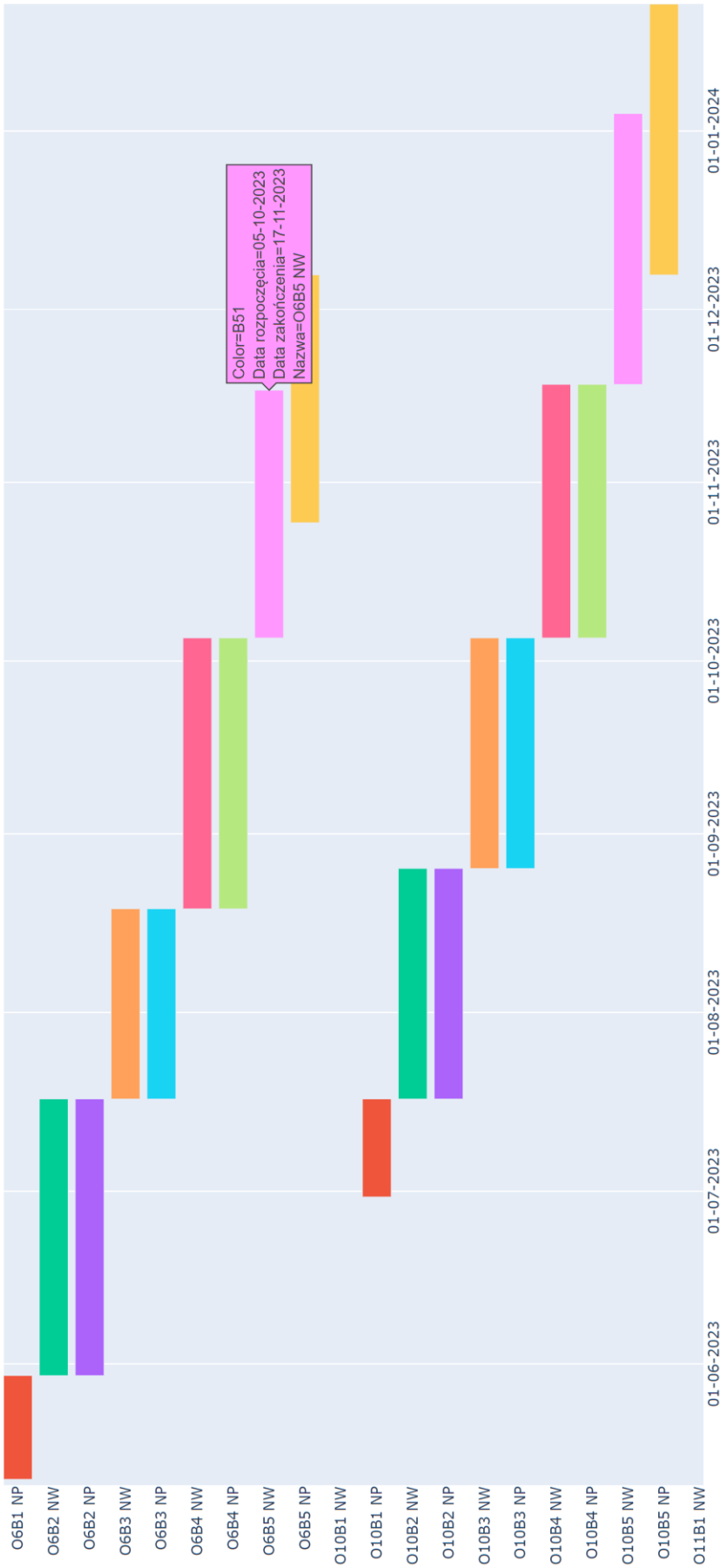
przedsięwzięcia – 1012, natomiast najniższe koszty całkowite uzyskała iteracja 1 i wynosiły one 8699,9.

Analizując uzyskane wyniki dla uszeregowania bazowego można zauważyć, że zarówno funkcja celu, koszty całkowite, czas realizacji oraz dotrzymanie ograniczeń znacząco odstaje od uzyskanych pięciu rozwiązań suboptymalnych. W porównaniu do iteracji numer 5 koszty całkowite różnią się o ok. 2,5%, czas realizacji o ok. 2,9%, a różnica w niedotrzymaniu ograniczeń wynosi 40.

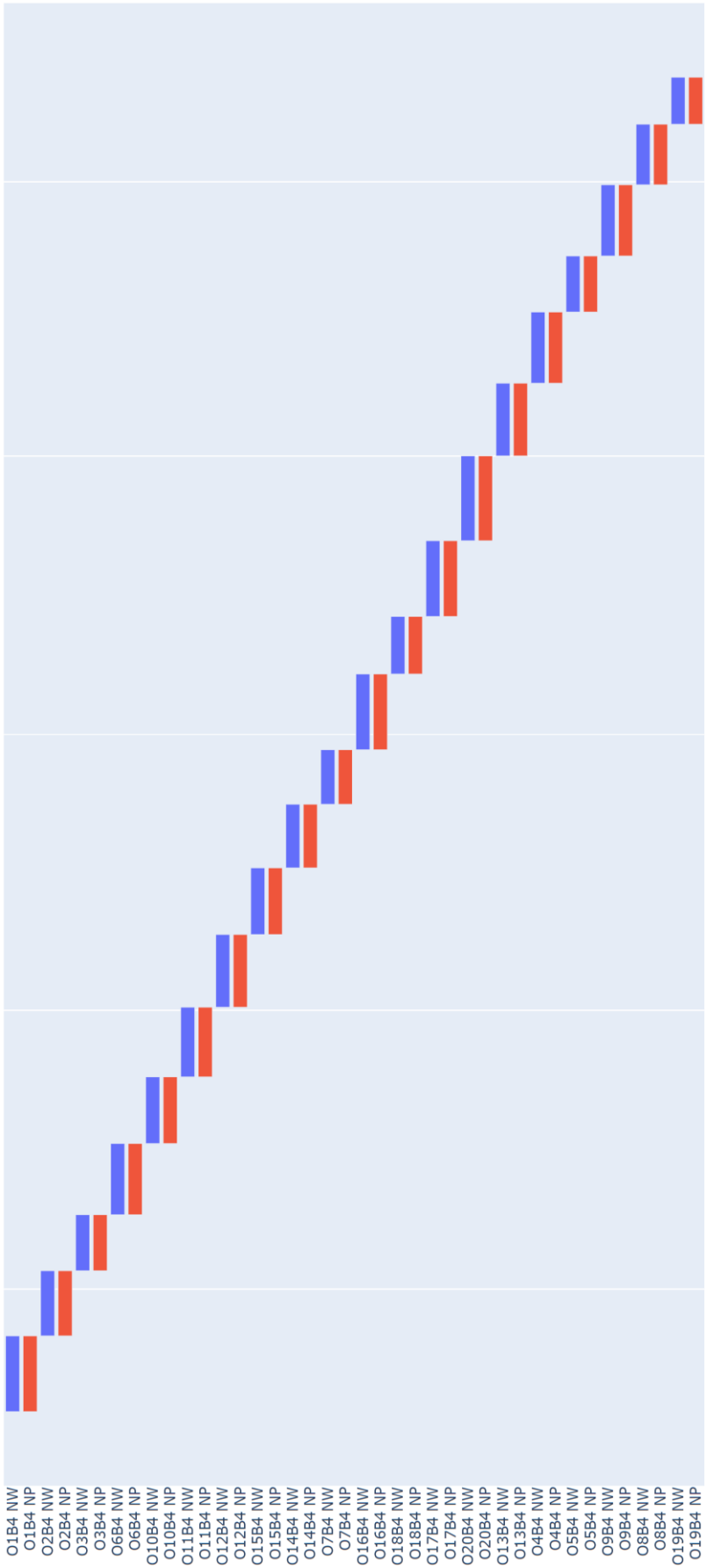
W tabeli 21 został przedstawiony przykładowy przebieg algorytmu. Jest to dokładnie przebieg iteracji numer 5. Widać na nim, że funkcja celu maleje wraz z kolejnymi iteracjami oraz wszystkie ograniczenia związane z kolejnością wykonania obiektów są dotrzymane. Na początku funkcja celu przyjmuje wysokie wartości, co jest spowodowane niedotrzymaniem ograniczeń związanych z harmonogramowaniem priorytetowym. Od iteracji 37277 zostało znalezione rozwiązanie, które dotrzymuje wszystkie ograniczenia harmonogramowania priorytetowego, a następnie są minimalizowane koszty realizacji przedsięwzięcia. Dla tej iteracji suboptymalne rozwiązanie o funkcji celu równej 9035,6 osiągnięto dla uszeregowania: [1, 2, 3, 6, 10, 11, 12, 15, 14, 7, 16, 18, 17, 20, 13, 4, 5, 9, 8, 19]. Dla otrzymanego uszeregowania wygenerowano z zastosowaniem modułu użytkowego harmonogram oraz dane wejściowe do programu do harmonogramowania. Na rysunku 29 przedstawiono wykres Gantta dla fragmentu przedsięwzięcia pomiędzy datami 12.05.2023 a 23.01.2024 (wykres Gantta dla całego przedsięwzięcia byłby nieczytelny). Na rysunku 30 przedstawiono fragment wykresu Gantta, ograniczony tylko do terminów najwcześniejszych i najpóźniejszych dla brygady B4. Widać na tym wykresie zachowanie ciągłości pracy tej brygady. W tabeli 22 przedstawiono dane wyjściowe modułu użytkowego, które mogą być użyte do eksportu wyników do programu do harmonogramowania. Jako termin rozpoczęcia przedsięwzięcia wybrano 1.01.2023 r.

Tabela 21: Przykładowy przebieg algorytmu dla przykładu 1 (iteracja 5).

Czas działania programu	Funkcja celu	Węzeł	Iteracja	Uszeregowanie
0,108735085	298734,1	0	1	[1, 9, 8, 17, 19, 4, 5, 16, 10, 11, 12, 2, 3, 18, 13, 6, 20, 7, 15, 14]
0,222001314	238741,4	0	2	[1, 12, 11, 10, 17, 9, 8, 20, 6, 13, 2, 3, 19, 16, 4, 5, 7, 15, 14, 18]
0,32971406	198692,1	0	3	[1, 15, 8, 9, 18, 12, 10, 11, 2, 3, 14, 17, 20, 6, 4, 5, 13, 16, 7, 19]
0,783027887	189053,1	0	7	[1, 11, 12, 10, 18, 9, 8, 15, 13, 16, 2, 3, 14, 4, 5, 19, 20, 17, 6, 7]
1,121121407	38878,5	0	10	[1, 9, 8, 12, 10, 11, 14, 6, 18, 16, 4, 5, 15, 13, 20, 17, 2, 3, 19, 7]
11,6968503	29418,9	0	107	[1, 2, 3, 6, 10, 12, 11, 14, 18, 4, 5, 8, 9, 15, 20, 16, 13, 19, 17, 7]
5040,100789	28955,8	5	37014	[1, 2, 3, 6, 10, 11, 12, 15, 14, 8, 9, 13, 7, 20, 18, 17, 4, 5, 16, 19]
7534,591225	9400,6	9	37277	[1, 2, 3, 6, 10, 11, 12, 15, 14, 7, 13, 18, 4, 5, 16, 17, 20, 9, 8, 19]
7535,402056	9166,2	9	37284	[1, 2, 3, 6, 10, 11, 12, 15, 14, 7, 16, 17, 20, 18, 8, 9, 4, 5, 13, 19]
7535,512759	9035,6	9	37285	[1, 2, 3, 6, 10, 11, 12, 15, 14, 7, 16, 18, 17, 20, 13, 4, 5, 9, 8, 19]



Rysunek 29: Fragment wykresu Gantta dla przykładu 1 (iteracja 5) pomiędzy datami 12.05.2023 a 23.01.2024.



Rysunek 30: Harmonogram pochodny dla brygady B4 dla przykładu 1 (iteracja 5).

Tabela 22: Dane wyjściowe modułu użytkowego dla przykładu 1 (iteracja 5).

Nazwa	Najwcześniejsze rozpoczęcie	Najwcześniejsze zakończenie	Najpóźniejsze rozpoczęcie	Najpóźniejsze zakończenie
O1B1	2023-01-01	2023-01-12	2023-01-01	2023-01-12
O1B2	2023-01-12	2023-02-24	2023-01-12	2023-02-24
O1B3	2023-02-24	2023-04-09	2023-02-26	2023-04-11
O1B4	2023-04-11	2023-05-31	2023-04-11	2023-05-31
O1B5	2023-05-31	2023-07-12	2023-06-30	2023-08-11
O2B1	2023-01-12	2023-01-31	2023-02-05	2023-02-24
O2B2	2023-02-24	2023-04-12	2023-02-24	2023-04-12
O2B3	2023-04-12	2023-05-21	2023-04-22	2023-05-31
O2B4	2023-05-31	2023-07-13	2023-05-31	2023-07-13
O2B5	2023-07-13	2023-08-13	2023-08-11	2023-09-11
O3B1	2023-01-31	2023-02-20	2023-03-23	2023-04-12
O3B2	2023-04-12	2023-05-30	2023-04-12	2023-05-30
O3B3	2023-05-30	2023-07-07	2023-06-05	2023-07-13
O3B4	2023-07-13	2023-08-19	2023-07-13	2023-08-19
O3B5	2023-08-19	2023-10-02	2023-09-11	2023-10-25
O6B1	2023-02-20	2023-03-10	2023-05-12	2023-05-30
O6B2	2023-05-30	2023-07-17	2023-05-30	2023-07-17
O6B3	2023-07-17	2023-08-19	2023-07-17	2023-08-19
O6B4	2023-08-19	2023-10-05	2023-08-19	2023-10-05
O6B5	2023-10-05	2023-11-17	2023-10-25	2023-12-07
O10B1	2023-03-10	2023-03-27	2023-06-30	2023-07-17
O10B2	2023-07-17	2023-08-26	2023-07-17	2023-08-26
O10B3	2023-08-26	2023-10-05	2023-08-26	2023-10-05
O10B4	2023-10-05	2023-11-18	2023-10-05	2023-11-18
O10B5	2023-11-18	2024-01-04	2023-12-07	2024-01-23
O11B1	2023-03-27	2023-04-18	2023-08-04	2023-08-26
O11B2	2023-08-26	2023-10-10	2023-08-26	2023-10-10
O11B3	2023-10-10	2023-11-17	2023-10-11	2023-11-18
O11B4	2023-11-18	2024-01-03	2023-11-18	2024-01-03
O11B5	2024-01-04	2024-02-11	2024-01-23	2024-03-01
O12B1	2023-04-18	2023-05-07	2023-09-21	2023-10-10
O12B2	2023-10-10	2023-11-22	2023-10-10	2023-11-22
O12B3	2023-11-22	2024-01-03	2023-11-22	2024-01-03
O12B4	2024-01-03	2024-02-20	2024-01-03	2024-02-20
O12B5	2024-02-20	2024-03-26	2024-03-01	2024-04-05
O15B1	2023-05-07	2023-05-27	2023-11-02	2023-11-22
O15B2	2023-11-22	2024-01-05	2023-11-22	2024-01-05
O15B3	2024-01-05	2024-02-12	2024-01-13	2024-02-20
O15B4	2024-02-20	2024-04-04	2024-02-20	2024-04-04
O15B5	2024-04-04	2024-05-15	2024-04-05	2024-05-16
O14B1	2023-05-27	2023-06-19	2023-12-13	2024-01-05
O14B2	2024-01-05	2024-02-18	2024-01-05	2024-02-18
O14B3	2024-02-18	2024-04-01	2024-02-21	2024-04-04
O14B4	2024-04-04	2024-05-16	2024-04-04	2024-05-16
O14B5	2024-05-16	2024-07-13	2024-05-16	2024-07-13
O7B1	2023-06-19	2023-07-16	2024-01-22	2024-02-18
O7B2	2024-02-18	2024-04-02	2024-02-18	2024-04-02
O7B3	2024-04-02	2024-05-07	2024-04-11	2024-05-16
O7B4	2024-05-16	2024-06-21	2024-05-16	2024-06-21
O7B5	2024-07-13	2024-09-07	2024-07-13	2024-09-07
O16B1	2023-07-16	2023-08-07	2024-03-11	2024-04-02
O16B2	2024-04-02	2024-05-17	2024-04-02	2024-05-17
O16B3	2024-05-17	2024-06-21	2024-05-17	2024-06-21
O16B4	2024-06-21	2024-08-10	2024-06-21	2024-08-10
O16B5	2024-09-07	2024-10-08	2024-09-07	2024-10-08
O18B1	2023-08-07	2023-08-26	2024-04-28	2024-05-17
O18B2	2024-05-17	2024-07-02	2024-05-17	2024-07-02
O18B3	2024-07-02	2024-08-10	2024-07-02	2024-08-10
O18B4	2024-08-10	2024-09-17	2024-08-10	2024-09-17
O18B5	2024-10-08	2024-12-05	2024-10-08	2024-12-05
O17B1	2023-08-26	2023-09-16	2024-06-11	2024-07-02
O17B2	2024-07-02	2024-08-15	2024-07-02	2024-08-15
O17B3	2024-08-15	2024-09-17	2024-08-15	2024-09-17
O17B4	2024-09-17	2024-11-06	2024-09-17	2024-11-06
O17B5	2024-12-05	2025-01-22	2024-12-05	2025-01-22
O20B1	2023-09-16	2023-10-11	2024-07-21	2024-08-15
O20B2	2024-08-15	2024-09-30	2024-08-15	2024-09-30
O20B3	2024-09-30	2024-11-06	2024-09-30	2024-11-06
O20B4	2024-11-06	2025-01-01	2024-11-06	2025-01-01
O20B5	2025-01-22	2025-03-14	2025-01-22	2025-03-14
O13B1	2023-10-11	2023-10-31	2024-09-10	2024-09-30
O13B2	2024-09-30	2024-11-15	2024-09-30	2024-11-15
O13B3	2024-11-15	2024-12-29	2024-11-18	2025-01-01
O13B4	2025-01-01	2025-02-18	2025-01-01	2025-02-18
O13B5	2025-03-14	2025-04-20	2025-03-14	2025-04-20
O4B1	2023-10-31	2023-11-23	2024-10-23	2024-11-15
O4B2	2024-11-15	2024-12-31	2024-11-15	2024-12-31
O4B3	2024-12-31	2025-02-13	2025-01-05	2025-02-18
O4B4	2025-02-18	2025-04-06	2025-02-18	2025-04-06
O4B5	2025-04-20	2025-05-21	2025-04-20	2025-05-21
O5B1	2023-11-23	2023-12-12	2024-12-12	2024-12-31
O5B2	2024-12-31	2025-02-18	2024-12-31	2025-02-18
O5B3	2025-02-18	2025-04-03	2025-02-20	2025-04-05
O5B4	2025-04-06	2025-05-13	2025-04-06	2025-05-13
O5B5	2025-05-21	2025-07-04	2025-05-21	2025-07-04
O9B1	2023-12-12	2024-01-07	2025-01-23	2025-02-18
O9B2	2025-02-18	2025-04-05	2025-02-18	2025-04-05
O9B3	2025-04-05	2025-05-13	2025-04-05	2025-05-13
O9B4	2025-05-13	2025-06-29	2025-05-13	2025-06-29
O9B5	2025-07-04	2025-08-08	2025-07-04	2025-08-08
O8B1	2024-01-07	2024-01-26	2025-02-18	2025-03-09
O8B2	2025-04-05	2025-05-22	2025-04-05	2025-05-22
O8B3	2025-05-22	2025-06-29	2025-05-22	2025-06-29
O8B4	2025-06-29	2025-08-08	2025-06-29	2025-08-08
O8B5	2025-08-08	2025-09-08	2025-08-08	2025-09-08
O19B1	2025-05-06	2025-05-22	2025-05-06	2025-05-22
O19B2	2025-05-22	2025-07-08	2025-05-22	2025-07-08
O19B3	2025-07-08	2025-08-08	2025-07-08	2025-08-08
O19B4	2025-08-08	2025-09-08	2025-08-08	2025-09-08
O19B5	2025-09-08	2025-10-09	2025-09-08	2025-10-09

4.5.2. Przykład 2

Przykład 2 polega na realizacji 3 budynków. Każdy z nich składa się z 3 kondygnacji, każda kondygnacja jest podzielona na 3 działki robocze. Schematyczny widok boczny został przedstawiony na rysunku 31. Cała realizacja składa się z 27 działek roboczych, które będą zgodnie z przyjętym nazewnictwem – obiektami.

	Budynek 1			Budynek 2			Budynek 3		
Kondygnacja 3	O7	O8	O9	O16	O17	O18	O25	O26	O27
Kondygnacja 2	O4	O5	O6	O13	O14	O15	O22	O23	O24
Kondygnacja 1	O1	O2	O3	O10	O11	O12	O19	O20	O21

Rysunek 31: Schematyczny widok boczny dla przykładu 2.

Na każdym obiekcie są do wykonania 4 procesy przez 4 wyspecjalizowane brygady. Wszystkie parametry zostały wygenerowane w taki sam sposób, jak w przykładzie 1. Wszystkie dane znajdują się w załączniku 15. W tym przykładzie wzięto pod uwagę tylko ograniczenia wynikające z metody CPM, natomiast wzięto pod uwagę ograniczenia kolejności wykonania obiektów:

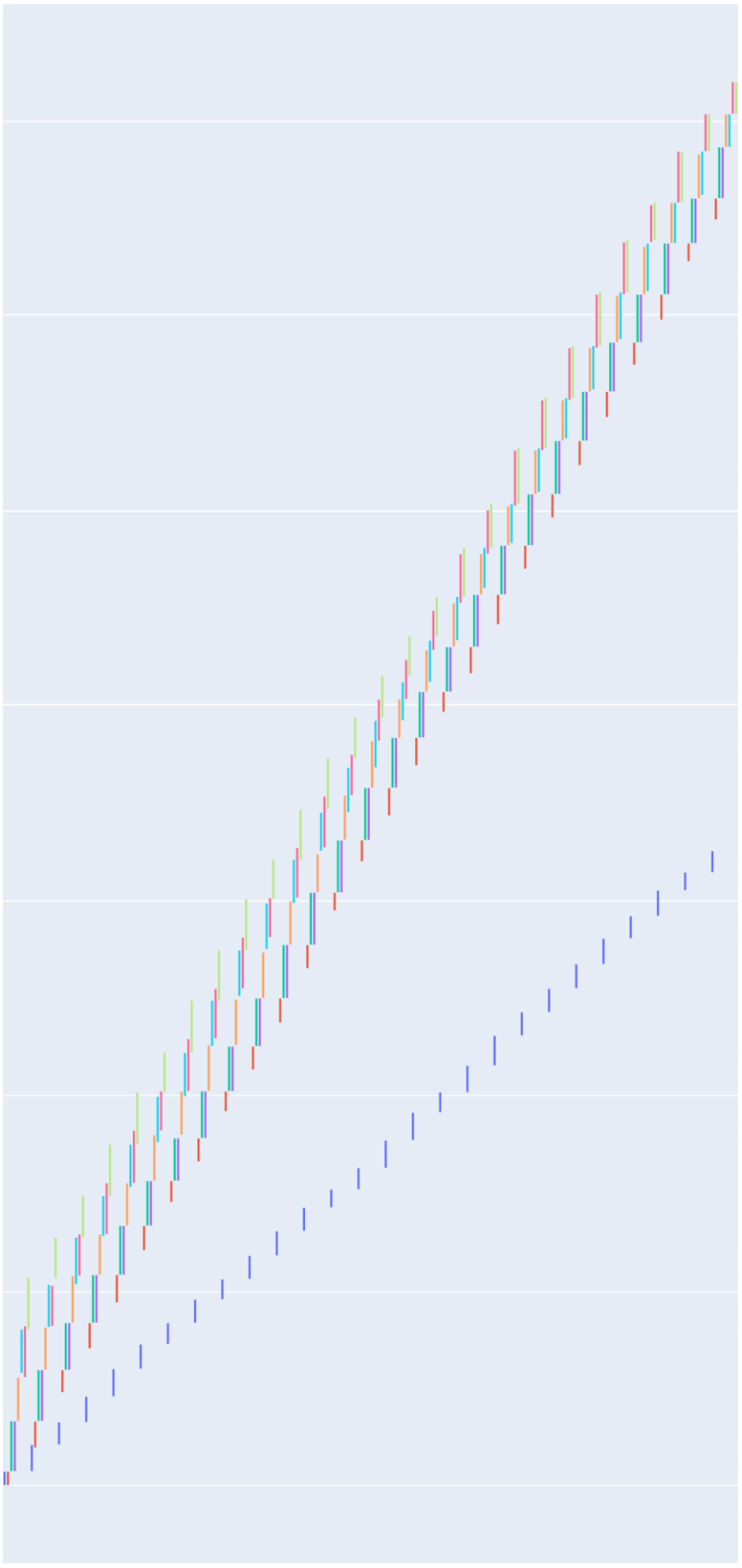
- Każda kondygnacja powinna być realizowana jako całość (np. O1, O2, O3 realizowane po sobie i nie mogą być przedzielone innym obiektem);
- W ramach jednego budynku realizacja kondygnacji musi następować po sobie (np. kondygnacja 2 po 1);
- Kolejność realizacji budynków jest dowolna (brygada może przechodzić z obiektu na obiekt, jeżeli zrealizuje daną kondygnację).

Dla takich ograniczeń kolejności wykonania obiektów, liczba możliwych realizacji przedsięwzięcia wynosi ok. $1{,}7 \cdot 10^{10}$. W module optymalizacji dyskretnej przyjęto takie same parametry, jak w przykładzie 1. Obliczenia wykonano dla 5 iteracji algorytmu. W tabeli 23 przedstawiono uzyskane wyniki wraz z uszeregowaniem początkowym (oznaczonym jako iteracja 0). Ze względu na brak dodatkowych ograniczeń harmonogramowania priorytetowego, uzyskiwana wartość funkcji celu jest praktycznie równa uzyskanym kosztom całkowitym, dlatego nie pokazano też stopnia niedotrzymania ograniczeń jak w przykładzie 1. Dla uszeregowania początkowego koszty całkowite wynosiły 10226,5 a czas 1318. Dla iteracji 4 uzyskano najlepsze pod względem kosztów uszeregowanie [12, 11, 10, 14, 13, 15, 17, 18, 16, 2, 3, 1, 4, 5, 6, 9, 8, 7, 20, 21, 19, 23, 24, 22, 25, 26, 27]. Dla takiego uszeregowania uzyskano koszt realizacji na poziomie 100029,3 oraz czas 1314. Daje to wynik lepszy o ok. 1,9% jeżeli chodzi o koszt i ok. 0,3% jeżeli chodzi o czas realizacji przedsięwzięcia. Należy jeszcze zwrócić uwagę, że wszystkie otrzymane uszeregowania spełniają założenia związane z kolejnością wykonania obiektów.

Tabela 23: Wyniki uzyskane dla pięciu iteracji (oraz uszeregowania początkowego) wykonane dla przykładu 2.

Iteracja	Uszeregowanie	Koszty całkowite ≈ Funkcja celu	Czas realizacji [dni]
0	[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27]	10226,5	1318
1	[12,11,10,15,13,14,16,18,17,21,19,20,22,24,23,2,3,1,5,6,4,9,8,7,25,26,27]	10054,0	1314
2	[12,10,11,14,13,15,16,18,17,3,1,2,5,4,6,8,7,9,20,21,19,24,23,22,25,26,27]	10034,9	1314
3	[10,12,11,1,3,2,13,15,14,16,18,17,5,6,4,8,7,9,20,19,21,24,23,22,25,26,27]	10121,7	1318
4	[12,11,10,14,13,15,17,18,16,2,3,1,4,5,6,9,8,7,20,21,19,23,24,22,25,26,27]	10029,3	1314
5	[12,10,11,15,13,14,16,17,18,3,2,1,21,19,20,22,23,24,5,6,4,9,8,7,25,26,27]	10042,0	1314

Na rysunku 32 zaprezentowano uzyskany wykres Gantta dla iteracji 4. Jest to widok ogólny całego harmonogramu wraz z terminami najwcześniejszymi i najpóźniejszymi. Ze względu na liczbę procesów do wykonania (27 obiektów razy 4 procesy na obiekcie = 108 procesów), pokazanie dokładnych danych (dat, terminów rozpoczęcia, zakończenia) na takim harmonogramie jest nieczytelne, natomiast autor chciał zaprezentować końcowy wynik jaki można uzyskać, stosując opracowaną metodę i możliwości modułu użytkowego. Oczywiście istnieje możliwość wygenerowania tabeli terminów rozpoczęcia i zakończenia, zarówno najwcześniejszych i najpóźniejszych, jednak taką możliwość pokazano już w przykładzie 1. Jako początek przedsięwzięcia ustalono 1 stycznia 2023 r.



Rysunek 32: Widok ogólny wykresu Gantta dla przykładu 2 (iteracja 4).

4.5.3. Przykład 3

W przykładzie 3 przeanalizowano taki sam przypadek jak w przykładzie 2 z tą różnicą, że przyjęto dodatkowe ograniczenia technologiczno-organizacyjne. Ten przykład składa się również z 27 obiektów, gdzie należy wykonać 4 procesy przez 4 wyspecjalizowane brygady robocze. Wszystkie dane liczbowe (koszty, czasy, terminy dyrektywne, kary za niedotrzymanie terminów) są takie same jak w przykładzie 2. Takie same są również ograniczenia na kolejność wykonania obiektów.

Dodatkowym ograniczeniem technologiczno-organizacyjnym jest zapewnienie ciągłości pracy dla brygady B2 oraz dla brygady B3. Ograniczenie to jest najważniejszym ograniczeniem, ważniejszym niż zapewnienie warunków metody CPM. Zapewnienie ciągłości pracy brygady B2 i B3 jest tak samo ważne. Kod programu realizujący ten przykład przedstawiono w załączniku 16.

Obliczenia wykonano, jak w pozostałych przykładach, dla pięciu iteracji. Wyniki przedstawiono w tabeli 24 (również z iteracją początkową oznaczoną jako iteracja 0). W każdym z przykładów udało się dochować ograniczeń technologiczno-organizacyjnych (ciągłości pracy brygad B2 i B3), dlatego wartość funkcji celu była zbliżona do wartości kosztów całkowitych. Z tego powodu nie pokazano w tabeli stopnia niedotrzymania ograniczeń technologiczno-organizacyjnych.

Tabela 24: Wyniki uzyskane dla pięciu iteracji (oraz uszeregowania początkowego) wykonane dla przykładu 3.

Iteracja	Uszeregowanie	Koszty całkowite ≈ Funkcja celu	Czas realizacji [dni]
0	[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17, 18,19,20,21,22,23,24,25,26,27]	12996,5	1340
1	[12,11,10,1,3,2,20,19,21,6,4,5,8,7,9,23,24, 22,14,13,15,17,16,18,26,25,27]	12319,1	1326
2	[1,2,3,19,21,20,6,4,5,8,7,9,11,12,10,13,15, 14,23,22,24,16,17,18,26,25,27]	12438,3	1333
3	[19,21,20,2,1,3,6,5,4,8,7,9,11,12,10,14,13, 15,17,16,18,22,23,24,26,25,27]	12296,8	1328
4	[2,3,1,5,4,6,20,19,21,9,7,8,10,12,11,23,24, 22,13,14,15,17,16,18,26,25,27]	12454,2	1329
5	[21,19,20,1,3,2,5,6,4,9,7,8,12,11,10,15,14, 13,23,24,22,17,16,18,26,25,27]	12531,7	1329

Dla uszeregowania początkowego otrzymano koszty na poziomie 12996,5 oraz czas realizacji 1340 dni. Najlepsze rozwiązanie udało się uzyskać dla iteracji 3, czyli uszeregowania [19,21,20,2,1,3,6,5,4,8,7,9,11,12,10,14,13,15,17,16,18,22,23,24,26,25,27]. Dla tej iteracji otrzymano koszt całkowity na poziomie 12296,8 oraz czas 1328 dni. Uzyskano poprawę w stosunku do iteracji 0 na poziomie ok. 5,4% jeżeli chodzi o koszt oraz ok. 0,9% jeżeli chodzi o czas realizacji. Należy zwrócić uwagę, że dla iteracji 0 również wykonano optymalizację czasowo-kosztowym modelem priorytetowego harmonogramowania. Każde uzyskane rozwiązanie jest lepsze zarówno jeżeli chodzi o koszty, jak i czas realizacji w stosunku do iteracji 0. Dodatkowo

każde uzyskane uszeregowanie spełnia narzucone ograniczenia na kolejność wykonania obiektów.

W tabeli 25 zaprezentowano porównanie wyników uzyskanych dla przykładu 2 i dla przykładu 3. Należy przypomnieć, że są to identyczne przykłady, natomiast przykład 3 posiada dodatkowe ograniczenia technologiczno-organizacyjne (ciągłość pracy brygady 2 i 3). Dokonano dodatkowych obliczeń dla uzyskanych rozwiązań suboptymalnych przy uwzględnieniu ograniczeń z drugiego przykładu.

Pierwszy wiersz tabeli prezentuje rozwiązanie suboptymalne uzyskane dla przykładu 2. Dla tego uszeregowania wykonano obliczenia, uwzględniając ograniczenia technologiczno-organizacyjne z przykładu 3. Uzyskano wartość kosztów na poziomie 13827,0. Dokonano takiej analizy krzyżowej dla wszystkich $2 \cdot 5$ iteracji. Można zaobserwować, zgodnie z oczekiwaniami, że uszeregowania otrzymane dla przykładu 3, uzyskują gorsze wyniki, jeżeli obłoży się je takimi samymi ograniczeniami, jak w przykładzie 2 (średni wynik 10246,94 jest gorszy niż 10056,38 – różnica ok. 1,9%). Analogicznie uszeregowania otrzymane dla przykładu 2, uzyskują gorsze wyniki, jeżeli doda się ograniczenia technologiczno-organizacyjne z przykładu 3 (średni wynik 13373,74 jest gorszy niż 12531,7 – różnica ok. 7,7%). Wynika z tego, że dodawanie albo usuwanie ograniczeń technologiczno-organizacyjnych wpływa na uzyskiwane suboptymalne rozwiązania. Oczywiście potwierdza to prawidłowe działanie opracowanej metody.

Tabela 25: Porównanie przykładu 2 i przykładu 3

	Iteracja	Uszeregowanie	Koszty – przykład 2	Koszty – przykład 3
Przykład 2	1	[12,11,10,15,13,14,16,18,17,21,19,20,22,24,23,2,3,1,5,6,4,9,8,7,25,26,27]	10054,0	13827,0
	2	[12,10,11,14,13,15,16,18,17,3,1,2,5,4,6,8,7,9,20,21,19,24,23,22,25,26,27]	10034,9	13109,8
	3	[10,12,11,1,3,2,13,15,14,16,18,17,5,6,4,8,7,9,20,19,21,24,23,22,25,26,27]	10121,7	13223,1
	4	[12,11,10,14,13,15,17,18,16,2,3,1,4,5,6,9,8,7,20,21,19,23,24,22,25,26,27]	10029,3	13046,8
	5	[12,10,11,15,13,14,16,17,18,3,2,1,21,19,20,22,23,24,5,6,4,9,8,7,25,26,27]	10042,0	13662,0
		Średnia	10056,38	13373,74
Przykład 3	1	[12,11,10,1,3,2,20,19,21,6,4,5,8,7,9,23,24,22,14,13,15,17,16,18,26,25,27]	10202,7	12319,1
	2	[1,2,3,19,21,20,6,4,5,8,7,9,11,12,10,13,15,14,23,22,24,16,17,18,26,25,27]	10279,6	12438,3
	3	[19,21,20,2,1,3,6,5,4,8,7,9,11,12,10,14,13,15,17,16,18,22,23,24,26,25,27]	10224,3	12296,8
	4	[2,3,1,5,4,6,20,19,21,9,7,8,10,12,11,23,24,22,13,14,15,17,16,18,26,25,27]	10260,3	12454,2
	5	[21,19,20,1,3,2,5,6,4,9,7,8,12,11,10,15,14,13,23,24,22,17,16,18,26,25,27]	10262,8	12531,7
		Średnia	10245,94	12408,02

Analizując uzyskane rezultaty dla wszystkich trzech przykładów (łącznie 15 iteracji) można zauważyć, że zaproponowana metoda pozwala na zredukowanie kosztów całkowitych przedsięwzięcia średnio o ok. 3,6%, przy możliwie największym stopniu dotrzymania ograniczeń

technologiczno-organizacyjnych. Pozwala również na zredukowanie czasu realizacji przedsięwzięcia średnio o ok. 0,7%.

4.6. Weryfikacja i walidacja opracowanego modelu

Opracowany model został poddany weryfikacji i walidacji zgodnie z wytycznymi zawartymi w publikacji (Robert G. Sargent, 2010). Przedstawiono w niej cztery różne podejścia do weryfikacji i walidacji modeli symulacyjnych.

4.6.1. Walidacja konceptualna modelu

Teorie i założenia zostały odpowiednio dobrane. Teorie takie jak programowania liniowe, metoda sprzężeń czasowych, czy MCTS są dobrze rozpoznane i wykorzystywane w podobnych do analizowanych problemach, co zostało poparte przeglądem literatury. Dodatkowo zostały zastosowane elementy analizy statystycznej. W zależności od wielkości, zadania model wykazywał wysoką średnią skuteczność w odnajdywaniu rozwiązania optymalnego oraz małą wartość odchylenia standardowego. Analiza schematu blokowego oraz zestawu ograniczeń wykazała poprawność modelu. Zastosowano tzw. walidację śledzenia, czyli sprawdzono skuteczność i poprawność modeli cząstkowych (model kosztowy zlinearyzowany, harmonogramowanie priorytetowe, optymalizacja z zastosowaniem MCTS) oraz modelu kompleksowego (na trzech quasiprawdziwych przykładach).

4.6.2. Skomputeryzowana weryfikacja modelu

Wszystkie moduły zostały zaimplementowane przez autora w języku programowania Python z zastosowaniem sprawdzonych ogólnodostępnych bibliotek (takich jak PyMathProg, NumPy, Pandas). Kod programu na etapie powstawania był wielokrotnie sprawdzany i weryfikowana była poprawność jego działania.

4.6.3. Walidacja operacyjna modelu

Walidacja operacyjna określa, czy uzyskane wyniki mają dokładność wymaganą dla zamierzonego celu w zakresie zamierzonej stosowalności modelu. Walidacji operacyjnej został poddany każdy opracowany model oraz moduł.

Zlinearyzowany moduł czasowo-kosztowy został sprawdzony w rozdziale 4.1. Dla danego przykładu model zachowuje się zgodnie z oczekiwaniami, zwracając coraz niższe koszty przy rosnącym terminie dyrektywnym. Dodatkowo wyznacza optymalne koszty całkowite przy uwzględnieniu kosztów pośrednich.

Weryfikacja operacyjna modelu harmonogramowania priorytetowego nastąpiła przez analizę przykładów zaprezentowanych w rozdziale 4.2, realizujących założenia decyzyjne opisane w rozdziale 3.4. Wszystkie uzyskiwane wyniki zgadzały się z ograniczeniami i oczekiwaniami.

Dodatkowo dla modelu harmonogramowania priorytetowego (z uwzględnieniem zlinearyzowanego modułu czasowo kosztowego) wykonano analizę wrażliwości dla trzech przypadków. Pierwszy to suboptymalne rozwiązane uzyskane w rozdziale 4.5.1. Wzięto pod uwagę rozwiązanie uzyskane w piątej iteracji, czyli: [1, 2, 3, 6, 10, 11, 12, 15, 14, 7, 16, 18, 17, 20, 13, 4, 5, 9, 8, 19]. Drugi to suboptymalne rozwiązanie uzyskane w rozdziale 4.5.2. Wzięto pod uwagę rozwiązanie uzyskane w czwartej iteracji, czyli: [12, 11, 10, 14, 13, 15, 17, 18, 16, 2, 3, 1, 4, 5, 6, 9, 8, 7, 20, 21, 19, 23, 24, 22, 25, 26, 27]. Trzeci to suboptymalne rozwiązanie uzyskane w rozdziale 4.5.3. Czyli uszeregowanie: [19, 21, 20, 2, 1, 3, 6, 5, 4, 8, 7, 9, 11, 12, 10, 14, 13, 15, 17, 16, 18, 22, 23, 24, 26, 25, 27] uzyskane dla iteracji trzeciej. Pozostałe parametry były dokładnie takie jak w odpowiednich przykładach.

Uzyskane wyniki zostały przedstawione w tabelach 26-28. Analizę przeprowadzono dla czterech różnych wartości zmiany parametrów: -10%, -1%, +1% oraz +10%. Każdy parametr opisany w tabelach reprezentuje w rzeczywistości grupę parametrów. Dla przykładu parametr „Czas graniczny" oznacza czasy graniczne dla wszystkich procesów, czyli $(n \cdot m)$ parametrów. W tabelach pokazano, jak zmienia się wartość kosztu całkowitego oraz czasu realizacji przedsięwzięcia. W tabeli 29 zaprezentowano średnie zmiany, a w tabeli 30 odchylenia standardowe, wartości czasu i kosztu dla wszystkich trzech przykładów.

Można zauważyć, że model jest najbardziej wrażliwy na zmianę parametru „czas graniczny". Skrócenie czasu granicznego wszystkich procesów o 10% powoduje zmniejszenie kosztów średnio o 16,76% oraz skrócenie czasu realizacji średnio o 9,74%, natomiast wydłużenie czasu granicznego procesów o 10% powoduje wzrost kosztów o średnio 46,98% oraz czasu o średnio 9,89%. Najbardziej wrażliwy na zmianę tego parametru okazał się przykład 3, dla którego wzrost czasu granicznego o 10% spowodował wzrost kosztów całkowitych o 60,90%. Wpływ parametru „czas graniczny" na wynik końcowy jest znaczny, ponieważ jest on powiązany bezpośrednio z czasem realizacji oraz pośrednio (poprzez koszty niedotrzymania terminów dyrektywnych oraz kosztem pośrednim) z kosztem realizacji. Drugim wrażliwym na zmianę parametrem jest termin dyrektywny. Zmniejszenie wartości parametru o 10% może spowodować wzrost kosztów (związanych z kosztem niedotrzymania terminów dyrektywnych) o średnio 40,54%, a najwięcej, dla przykładu 3, o nawet 49,85%. Wrażliwymi parametrami są też koszt graniczny oraz koszt pośredni. Ich wpływ na koszt całkowity przy zmianie wartości parametru o 10% może wynosić średnio odpowiednio 5,70% oraz 4,68%. Zmiana wartości

pozostałych parametrów w proponowanym zakresie ma niewielki wpływ na czas i koszt realizacji przedsięwzięcia.

Należy zwrócić uwagę, że zmiana wartości parametrów związanych z harmonogramowaniem priorytetowym (ostatni wiersz) nie ma wpływu na wynik czasu i kosztu realizacji w tym przypadku. W opisywanym przykładzie wszystkie ograniczenia narzucone przez decydenta zostały dotrzymane, więc zmiana parametrów harmonogramowania priorytetowego w zakresie +/-10% nie ma wpływu na uzyskiwane wyniki.

Na podstawie analizy odchylenia standardowego (tabela 30) można zauważyć, że wyniki uzyskiwane dla analizy wrażliwości są dość stabilne (odchylenie standardowe jest bliskie 0). Dla parametrów „czas graniczny" i „termin dyrektywny" odchylenie przyjmuje największe wartości (nawet 10,04%), co oznacza, że wpływ zmiany tych parametrów na czas i koszt jest szeroko rozrzucony wokół średniej.

Tabela 26: Analiza wrażliwości dla przykładu 1.

		Zmiana [%]							
	Zmiana parametru o:	-10,00		-1,00		1,00		10,00	
		Koszt	Czas	Koszt	Czas	Koszt	Czas	Koszt	Czas
Zmieniany parametr	czas graniczny	-14,80	-9,21	-2,66	-0,94	3,03	0,94	42,49	9,67
	koszt graniczny	-5,87	0,00	-0,59	0,00	0,59	0,00	5,87	0,00
	parametry czasu związane z zlinearyzowanym modelem czasowo-kosztowym	1,71	-0,20	0,11	-0,06	-0,11	0,00	-1,00	0,03
	parametry kosztu związane z zlinearyzowanym modelem czasowo-kosztowym	0,81	-0,20	0,08	0,00	-0,08	0,00	-0,83	0,40
	koszt pośredni	-4,49	0,30	-0,45	0,00	0,45	0,00	4,47	-0,20
	koszt niedotrzymania terminu dyrektywnego	-0,52	0,00	-0,05	0,00	0,05	0,00	0,52	0,00
	koszt przedwczesnego wykonania obiektu	0,05	0,00	0,01	0,00	-0,01	0,00	-0,05	-0,20
	koszt nieciągłości	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	termin dyrektywny	36,92	-0,30	2,51	0,00	-2,14	0,00	-10,34	0,40
	termin dostępności brygad, termin dostępności obiektów, parametry związane z harmonogramowaniem priorytetowym	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Tabela 27: Analiza wrażliwości dla przykładu 2.

		Zmiana [%]							
	Zmiana parametru o:	-10,00		-1,00		1,00		10,00	
		Koszt	Czas	Koszt	Czas	Koszt	Czas	Koszt	Czas
Zmieniany parametr	czas graniczny	-12,98	-10,00	-1,30	-1,00	1,42	1,00	37,57	10,00
	koszt graniczny	-6,19	0,00	-0,62	0,00	0,62	0,00	6,19	0,00
	parametry czasu związane z zlinearyzowanym modelem czasowo-kosztowym	0,76	0,00	0,08	0,00	-0,07	0,00	-0,73	0,00
	parametry kosztu związane z zlinearyzowanym modelem czasowo-kosztowym	0,94	0,00	0,09	0,00	-0,09	0,00	-0,95	0,00
	koszt pośredni	-5,24	0,00	-0,52	0,00	0,52	0,00	5,24	0,00
	koszt niedotrzymania terminu dyrektywnego	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	koszt przedwczesnego wykonania obiektu	0,48	0,00	0,05	0,00	-0,05	0,00	-0,49	0,00
	koszt nieciągłości	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	termin dyrektywny	34,86	0,00	0,97	0,00	-0,84	0,00	-8,44	0,00
	termin dostępności brygad, termin dostępności obiektów, parametry związane z harmonogramowaniem priorytetowym	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Tabela 28: Analiza wrażliwości dla przykładu 3.

		Zmiana [%]							
	Zmiana parametru o:	-10,00		-1,00		1,00		10,00	
		Koszt	Czas	Koszt	Czas	Koszt	Czas	Koszt	Czas
Zmieniany parametr	czas graniczny	-22,51	-10,00	-4,23	-1,00	5,08	1,00	60,90	10,00
	koszt graniczny	-5,05	0,00	-0,50	0,00	0,50	0,00	5,05	0,00
	parametry czasu związane z zlinearyzowanym modelem czasowo-kosztowym	6,19	0,00	0,61	0,00	-0,58	0,00	-5,54	0,00
	parametry kosztu związane z zlinearyzowanym modelem czasowo-kosztowym	0,57	0,00	0,06	0,00	-0,06	0,00	-0,58	0,00
	koszt pośredni	-4,32	0,00	-0,43	0,00	0,43	0,00	4,32	0,00
	koszt niedotrzymania terminu dyrektywnego	-1,24	0,00	-0,12	0,00	0,12	0,00	1,23	0,00
	koszt przedwczesnego wykonania obiektu	0,03	0,00	0,00	0,00	0,00	0,00	-0,03	0,00
	koszt nieciągłości	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	termin dyrektywny	49,85	0,00	3,97	0,00	-3,24	0,00	-15,11	0,00
	termin dostępności brygad, termin dostępności obiektów, parametry związane z harmonogramowaniem priorytetowym	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Tabela 29: Średnia zmiana wartości czasu i kosztu przy zmianie parametrów dla przykładów 1,2,3.

		Zmiana [%]							
	Zmiana parametru o:	-10,00		-1,00		1,00		10,00	
		Koszt	Czas	Koszt	Czas	Koszt	Czas	Koszt	Czas
Zmieniany parametr	czas graniczny	-16,76	-9,74	-2,73	-0,98	3,18	0,98	46,98	9,89
	koszt graniczny	-5,70	0,00	-0,57	0,00	0,57	0,00	5,70	0,00
	parametry czasu związane z zlinearyzowanym modelem czasowo-kosztowym	2,89	-0,07	0,26	-0,02	-0,25	0,00	-2,42	0,01
	parametry kosztu związane z zlinearyzowanym modelem czasowo-kosztowym	0,77	-0,07	0,08	0,00	-0,08	0,00	-0,79	0,13
	koszt pośredni	-4,68	0,10	-0,47	0,00	0,47	0,00	4,68	-0,07
	koszt niedotrzymania terminu dyrektywnego	-0,59	0,00	-0,06	0,00	0,06	0,00	0,58	0,00
	koszt przedwczesnego wykonania obiektu	0,19	0,00	0,02	0,00	-0,02	0,00	-0,19	-0,07
	koszt nieciągłości	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	termin dyrektywny	40,54	-0,10	2,48	0,00	-2,08	0,00	-11,30	0,13
	termin dostępności brygad, termin dostępności obiektów, parametry związane z harmonogramowaniem priorytetowym	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Tabela 30: Odchylenie standardowe zmiany wartości czasu i kosztu przy zmianie parametrów dla przykładów 1,2,3.

		Zmiana [%]							
	Zmiana parametru o:	-10,00		-1,00		1,00		10,00	
		Koszt	Czas	Koszt	Czas	Koszt	Czas	Koszt	Czas
Zmieniany parametr	czas graniczny	4,13	0,37	1,20	0,03	1,50	0,03	10,04	0,15
	koszt graniczny	0,48	0,00	0,05	0,00	0,05	0,00	0,48	0,00
	parametry czasu związane z zlinearyzowanym modelem czasowo-kosztowym	2,37	0,09	0,24	0,03	0,23	0,00	2,20	0,01
	parametry kosztu związane z zlinearyzowanym modelem czasowo-kosztowym	0,15	0,09	0,01	0,00	0,01	0,00	0,15	0,19
	koszt pośredni	0,40	0,14	0,04	0,00	0,04	0,00	0,40	0,09
	koszt niedotrzymania terminu dyrektywnego	0,51	0,00	0,05	0,00	0,05	0,00	0,51	0,00
	koszt przedwczesnego wykonania obiektu	0,21	0,00	0,02	0,00	0,02	0,00	0,21	0,09
	koszt nieciągłości	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
	termin dyrektywny	6,64	0,14	1,22	0,00	0,98	0,00	2,81	0,19
	termin dostępności brygad, termin dostępności obiektów, parametry związane z harmonogramowaniem priorytetowym	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Dla modułu optymalizacji dyskretnej przeprowadzono weryfikację operacyjną na dwa sposoby. Po pierwsze, dokonano analizy wyników uzyskiwanych dla różnych parametrów C i sposobów wyboru węzła w metodzie MCTS. Można zauważyć, że każda możliwa kombinacja parametrów i wyborów węzłów powoduje poprawę wartości funkcji celu. W dalszej analizie

został wybrany ten, który dawał najlepsze możliwe wyniki. Szczegółowe rezultaty przedstawiono w rozdziale 4.3.

Drugą metoda weryfikacji operacyjnej dla modułu optymalizacji dyskretnej było porównanie uzyskiwanych wyników z dwoma innymi metodami. Porównano wyniki uzyskiwane przez zastosowanie opracowanej wersji metody MCTS z metodą MC, metodą SA oraz najlepszymi znanymi wynikami. Obliczenia zostały wykonane 10 razy dla każdego z 6 przykładów. Uzyskane wyniki są porównywalne. Najlepsza okazała się metoda SA, druga MCTS, trzecia MC, jednak należy zwrócić uwagę, że analiza nie uwzględniała ograniczeń narzuconych na kolejność wykonania obiektów. Analiza uwzględniająca ograniczenia kolejności wykonania obiektów została przeprowadzona dla dwóch różnych przykładów i dała oczekiwane, zadowalające rezultaty. Całość analizy została przedstawiona w rozdziale 4.3.

Dodatkowo przeprowadzono weryfikację operacyjną modelu kompleksowego, uwzględniającego wszystkie opracowane moduły. Dokonano analizy wyników dla trzech różnych przykładów z zastosowaniem kompleksowego modelu. Szczegółowo przykłady zostały opisane w rozdziale 4.5.

4.6.4. Walidacja danych

Dane wejściowe na każdym etapie zostały sprawdzone pod kątem poprawności z założeniami. Poprawność wprowadzonych danych można prześledzić w załącznikach do rozprawy, gdzie znajdują się kody wykorzystywanych programów.

5. Podsumowanie

5.1. Wnioski końcowe

Harmonogramowanie przedsięwzięć o charakterze powtarzalnym, przedsięwzięć wieloobiektowych jest trudne i skomplikowane. Liczba ograniczeń, zarówno technologicznych, organizacyjnych, jak również ekonomicznych jest tak duża, że skuteczna analiza oraz wybór rozwiązań suboptymalnych bez użycia modeli matematycznych i obliczeń komputerowych jest niemożliwa. Z drugiej strony jest widoczny wyraźny trend wzrostowy liczby realizacji tego typu przedsięwzięć, a więc zapotrzebowanie na rozwiązania wspomagające planowanie takich inwestycji jest duże.

Opracowana przez autora metoda priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych okazała się efektywnym narzędziem do tworzenia harmonogramów przedsięwzięć, gdzie możliwa jest zmiana kolejności wykonania obiektów. Wykorzystując opracowaną metodę dla trzech przykładów udało się zredukować koszty całkowite przedsięwzięcia średnio o ok. 3,6%, przy możliwie największym stopniu dotrzymania ograniczeń technologiczno-organizacyjnych. Dodatkowo zredukowano również czas realizacji przedsięwzięcia średnio o ok. 0,7%.

Opracowane modele (zlinearyzowany model czasowo-kosztowy, model priorytetowego harmonogramowania oraz czasowo-kosztowy model priorytetowego harmonogramowania) uzyskują zadowalające wyniki w procesie optymalizacji kosztowej dochowując narzuconych przez decydenta ograniczeń technologiczno-organizacyjnych, uwzględniając koszty bezpośrednie, pośrednie, kary za niedotrzymanie terminów dyrektywnych, nagrody za wcześniejsze zrealizowanie obiektów oraz kary za nieciągłość pracy brygad. Opracowana zmodyfikowana metoda MCTS osiąga zadowalające rezultaty oraz pozwala wyznaczyć suboptymalną kolejność wykonania obiektów przy założonych ograniczeniach. Skuteczność opracowanej modyfikacji pozwala uzyskać wyniki lepsze o 2 punkty procentowe, w porównaniu do metody MC. Opracowana metoda, wraz z modelami została zweryfikowana uznanymi metodami weryfikacji. Została również przeprowadzona analiza wrażliwości, która udowodniła prawidłowe działanie opracowanego czasowo-kosztowego modelu priorytetowego harmonogramowania wraz ze wskazaniem parametrów, na których zmianę jest wrażliwa. Opracowany moduł użytkowy pozwala generować robocze harmonogramy oraz tabele z datami rozpoczęcia i zakończenia poszczególnych procesów, co nadaje charakter praktyczny opracowanej metodzie.

Poniżej zestawiono najważniejsze elementy stanowiące naukową wartość dodaną niniejszej rozprawy:

- opracowanie nowej koncepcji priorytetowego harmonogramowania;
- opracowanie zlinearyzowanego modelu czasowo-kosztowego;
- opracowanie czasowo-kosztowego modelu priorytetowego harmonogramowania;
- opracowanie zestawu wag realizującego różne założenia decyzyjne;
- opracowanie modelu optymalizacji dyskretnej uszeregowania obiektów wykorzystującego zmodyfikowaną metodę MCTS;
- opracowanie metody priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych;

Poniżej zestawiono najważniejsze elementy stanowiące praktyczną wartość dodaną niniejszej rozprawy:

- opracowanie modułu decydenta;
- opracowanie modułu użytkowego;
- implementacja komputerowa opracowanego modelu.

W ocenie autora potwierdzono postawioną we wstępnie tezę oraz zrealizowano postawione we wstępnie cele.

5.2. Kierunki dalszych badań

Opracowana metoda i modele mogą być dalej rozwijane i rozszerzane o kolejne funkcjonalności. Poniżej wskazano kierunki dalszych badań:

- modyfikacja modelu w celu uwzględnienia danych w postaci niedeterministycznej – probabilistycznej lub rozmytej;
- rozszerzenie metody oraz modelu o możliwość wykonywania konkretnych procesów przez więcej niż jedną wyspecjalizowaną brygadę roboczą;
- opracowanie skuteczniejszej metody optymalizacji dyskretnej uwzględniającej narzucone ograniczenia kolejności wykonania obiektów. W związku z przeprowadzonymi testami, dobrą propozycją jest zmodyfikowanie metody SA;
- udoskonalenie metody MCTS, która na podstawie wstępnego rozwiązania modelu optymalizacji dyskretnej zaproponuje optymalną wartość parametru C oraz dobór metody wyboru węzła. Dobór parametrów i metody wyboru węzła

powinien być dokonywany bez udziału osoby odpowiedzialnej za tworzenie harmonogramu. Proponuje się wykorzystanie metody XGBoost;

- opracowanie interfejsu granicznego użytkownika, aby obsługa programu stała się bardziej przystępna dla użytkownika i/lub integracja opracowanego modelu z programem do harmonogramowania jak Microsoft Project. Może być to zrealizowane jako wtyczka rozszerzająca możliwości programu.

6. Bibliografia

Adamiecki, K. (1909). Metoda wykreślna organizowania pracy zbiorowej w walcowni. *Przegląd Techniczny*, *17–20*.

Adamiecki, K. (1931). Harmonograf. *Przegląd Organizacji*, *4*, 137–139.

Adamiecki, K. (1938). *O istocie naukowej organizacji*. Koło Naukowej Organizacji Studentów Politechniki Warszawskiej.

Afanasev, V. (1977). *Modeli organizacjii robot*. LISI.

Afanasev, V. (1980). *Algoritmy formiravanija rasceta i iptimazacji metd organizacii robot*.

Al Sarraj, Z. M. (1990). Formal Development of Line-of-Balance Technique. *Journal of Construction Engineering and Management*, *116*(4), 689–704. https://doi.org/10.1061/(ASCE)0733-9364(1990)116:4(689)

Altuwaim, A., & El-Rayes, K. (2018). Minimizing duration and crew work interruptions of repetitive construction projects. *Automation in Construction*, *88*, 59–72. https://doi.org/10.1016/j.autcon.2017.12.024

Birrell, G. (1980). Construction Planning—Beyond the Critical Path. *Journal of the Construction Division*, *106*(3), 389–407.

Biruk, S., & Rzepecki, L. (2019). Scheduling Repetitive Construction Processes Using the Learning-Forgetting Theory. *IOP Conference Series: Materials Science and Engineering*, *471*, 112039. https://doi.org/10.1088/1757-899X/471/11/112039

Bożejko, W., Hejducki, Z., Rajba, P., & Wodecki, M. (2012). Algorytm memetyczny dla pewnego problemu potokowego w budownictwie. *Innowacje w zarządzaniu i inżynierii produkcji*, 251–262.

Bożejko, W., Hejducki, Z., Uchroński, M., & Wodecki, M. (2014). Solving Resource-constrained Construction Scheduling Problems with Overlaps by Metaheuristic. *JOURNAL OF CIVIL ENGINEERING AND MANAGEMENT*, *20*(5), 649–659. https://doi.org/10.3846/13923730.2014.906496

Bożejko, W., Hejducki, Z., & Wodecki, M. (2019). Flowshop scheduling of construction processes with uncertain parameters. *Archives of Civil and Mechanical Engineering*, *19*(1), 194–204. https://doi.org/10.1016/j.acme.2018.09.010

Budownictwo mieszkaniowe w okresie styczeń-grudzień 2021 roku. (2022, styczeń 21). Główny Urząd Statystyczny. https://stat.gov.pl/obszary-tematyczne/przemysl-budownictwo-srodki-trwale/budownictwo/budownictwo-mieszkaniowe-w-okresie-styczen-grudzien-2021-roku,5,122.html

Bukłaha, E. (2016). Planowanie i kontrola projektów z zastosowaniem linii równowagi (line of balance) / Planning and controlling of projects using the line of balance technique. *Management Forum*, *4(1)*. https://doi.org/10.15611/mf.2016.4(1).07

Chen, A., Liang, Y.-C., & Padilla, J. (2014). An Entropy-Based Upper Bound Methodology for Robust Predictive Multi-Mode RCPSP Schedules. *Entropy*, *16*(9), 5032–5067. https://doi.org/10.3390/e16095032

Dolabi, H. R. Z., & Afshar, A. (2016). Cost Optimization of Repetitive Projects with Varying Production Rates Using a Max-Min Ant System. *Construction Research Congress 2016*, 636–646. https://doi.org/10.1061/9780784479827.065

El-Rayes, K., & Kandil, A. (2005). Time-Cost-Quality Trade-Off Analysis for Highway Construction. *Journal of Construction Engineering and Management*, *131*(4), 477–486. https://doi.org/10.1061/(ASCE)0733-9364(2005)131:4(477)

Ezeldin, A. S., & Soliman, A. (2009). Hybrid Time-Cost Optimization of Nonserial Repetitive Construction Projects. *Journal of Construction Engineering and Management*, *135*(1), 42–55. https://doi.org/10.1061/(ASCE)0733-9364(2009)135:1(42)

Gantt, H. L. (1910). Work, wages, and profits. *The Engineering magazine*.

García-Nieves, J. D., Ponz-Tienda, J. L., Salcedo-Bernal, A., & Pellicer, E. (2018). The Multimode Resource-Constrained Project Scheduling Problem for Repetitive Activities in Construction Projects. *Computer-Aided Civil and Infrastructure Engineering*, *33*(8), 655–671. https://doi.org/10.1111/mice.12356

Gelly, S., & Silver, D. (2007). Combining online and offline knowledge in UCT. *Proceedings of the 24th international conference on Machine learning*, 273–280. https://doi.org/10.1145/1273496.1273531

Goldratt, E. (1990). *What is this thing called the Theory of Constraints?* North River Press.

Goldratt, E. (1997). *Critical Chain*. The North River Press.

Gouda, A., Hosny, O., & Nassar, K. (2017). Optimal crew routing for linear repetitive projects using graph theory. *Automation in Construction*, *81*, 411–421. https://doi.org/10.1016/j.autcon.2017.03.007

Harris, R. B., & Ioannou, P. G. (1998). Scheduling Projects with Repeating Activities. *Journal of Construction Engineering and Management*, *124*(4), 269–278. https://doi.org/10.1061/(ASCE)0733-9364(1998)124:4(269)

Hegazy, T. (2001). Critical Path Method–Line of Balance Model for Efficient Scheduling of Repetitive Construction Projects. *Transportation Research Record: Journal of the Transportation Research Board*, *1761*(1), 124–129. https://doi.org/10.3141/1761-16

Hegazy, T., & Wassef, N. (2001). Cost Optimization in Projects with Repetitive Nonserial Activities. *Journal of Construction Engineering and Management*, *127*(3), 183–191. https://doi.org/10.1061/(ASCE)0733-9364(2001)127:3(183)

Hejducki, Z. (2004). *Zarządzanie czasem w procesach budowlanych z zastosowaniem modeli macierzowych*. Oficyna Wydawnicza Politechniki Wrocławskiej.

Hejducki, Z., & Rogalska, M. (2011). *Time coupling methods. Construction scheduling and time/cost optimization*. Oficyna Wydawnicza Politechniki Wrocławskiej.

Hejducki, Z., & Rogalska, M. (2017). *Harmonogramowanie procesów budowlanych metodami sprzężeń czasowych*. Politechnika Lubelska.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. The MIT Press. https://doi.org/10.7551/mitpress/1090.001.0001

Informacja o sprzedaży mieszkań przez ATAL S.A. w latach 2015-2021. (b.d.). Pobrano 14 lipiec 2022, z https://atal.pl/dla-inwestorow/raporty/biezace.html

Jaśkowski, P. (2015). Repetitive construction processes scheduling using mixed-integer linear programming. *Budownictwo i Architektura*, *14*(2), 55–61.

Jaśkowski, P., & Biruk, S. (2014). Harmonogramowanie pracy brygad realizujących budowlane procesy powtarzalne. *Journal of Science of the Gen. Tadeusz Kosciuszko Military Academy of Land Forces*, *173*(3), 197–204. https://doi.org/10.5604/17318157.1143798

Jaworski, K. (2009). *Metodologia projektowania realizacji budowy*. PWN.

Johnson, D. B. (1977). Efficient Algorithms for Shortest Paths in Sparse Networks. *Journal of the ACM*, *24*(1), 1–13. https://doi.org/10.1145/321992.321993

Jooken, J., Leyman, P., Wauters, T., & de Causmaecker, P. (2023). Exploring search space trees using an adapted version of Monte Carlo tree search for combinatorial optimization problems. *Computers & Operations Research*, *150*, 106070. https://doi.org/10.1016/j.cor.2022.106070

Józewczyk, J. (2001). *Wybrane problemy podejmowania decyzji w kompleksach operacji*. Oficyna Wydawnicza Politechniki Wrocławskiej.

Karp, R. M. (1972). Reducibility among Combinatorial Problems. W *Complexity of Computer Computations* (s. 85–103). Springer US. https://doi.org/10.1007/978-1-4684-2001-2_9

Kelley, J., & Walker, M. (1989). The origins of CPM: a personal history. *PM Network*, *3*(2), 7–22.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by Simulated Annealing. *Science*, *220*(4598), 671–680. https://doi.org/10.1126/science.220.4598.671

Kocsis, L., & Szepesvári, C. (2006). *Bandit Based Monte-Carlo Planning* (s. 282–293). https://doi.org/10.1007/11871842_29

Kostrzewa, P., & Rogalska, M. (2019). Scheduling Construction Processes Using the Probabilistic Time Coupling Method III. *IOP Conference Series: Materials Science and Engineering*, *471*. https://doi.org/10.1088/1757-899X/471/11/112072

Kostrzewa-Demczuk, P. (2022). *Harmonogramowanie metodami sprzężeń czasowych w ujęciu probabilistycznym*. Politechnika Świętokrzyska.

Krawczyńska-Piechna, A. (2015). *Interaktywna metoda planowania robót betonowych z analizą efektywności wykorzystania deskowań systemowych*. Politechnika Warszawska.

Krzemiński, M. (2016a). Construction Scheduling and Stability of the Resulting Schedules. *Archives of Civil Engineering*, *62*(2), 89–100. https://doi.org/10.1515/ace-2015-0067

Krzemiński, M. (2016b). KASS v.2.2. scheduling software for construction. *MATEC Web of Conferences*, *86*, 05002. https://doi.org/10.1051/matecconf/20168605002

Krzemiński, M. (2016c). KASS v.2.2. Scheduling Software for Construction with Optimization Criteria Description. *Acta Physica Polonica A*, *130*(6), 1439–1442. https://doi.org/10.12693/APhysPolA.130.1439

Krzemiński, M. (2017). Optimization of Work Schedules Executed using the Flow Shop Model, Assuming Multitasking Performed by Work Crews. *Archives of Civil Engineering*, *63*(4), 3–19. https://doi.org/10.1515/ace-2017-0037

Kulejewski, J., Ibadov, N., Rosłon, J., & Zawistowski, J. (2021). Cash Flow Optimization for Renewable Energy Construction Projects with a New Approach to Critical Chain Scheduling. *Energies*, *14*(18), 5795. https://doi.org/10.3390/en14185795

Land, A. H., & Doig, A. G. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica*, *28*(3), 497–520. https://doi.org/10.2307/1910129

Li, Y.-Z., Pan, Q.-K., He, X., Sang, H.-Y., Gao, K.-Z., & Jing, X.-L. (2022). The distributed flowshop scheduling problem with delivery dates and cumulative payoffs. *Computers & Industrial Engineering*, *165*. https://doi.org/10.1016/j.cie.2022.107961

Lin, S.-W., & Ying, K.-C. (2016). Minimizing makespan for solving the distributed no-wait flowshop scheduling problem. *Computers & Industrial Engineering*, *99*, 202–209. https://doi.org/10.1016/j.cie.2016.07.027

Line of Balance Technology: A Graphic Method of Industrial Programming. (1962a). United States: Navy Department.

Line of Balance Technology: A Graphic Method of Industrial Programming. (1962b). Navy Department.

Liu, S.-S., & Wang, C.-J. (2008). Cost Optimization Model for Linear Scheduling Problems Considering Work Continuity. *2008 IEEE International Conference on Communications*, 5505–5509. https://doi.org/10.1109/ICC.2008.1032

Lu, C., Chiu, S. Y., Wu, J., & Chao, L. P. (2016). Dynamic monte-carlo tree search algorithm for multi-objective flexible job-shop scheduling problem. *Applied Mathematics and Information Sciences*, *10*(4), 1531–1539. https://doi.org/10.18576/amis/100431

Lubosch, M., Kunath, M., & Winkler, H. (2018). Industrial scheduling with Monte Carlo tree search and machine learning. *Procedia CIRP*, *72*, 1283–1287. https://doi.org/10.1016/j.procir.2018.03.171

Marcinkowski, R. (2002). *Metody rozdziału zasobów realizatora w działalności inżynieryjno-budowlanej*. Wojskowa Akademia Techniczna.

Marcinkowski, R. (2017). Modelowanie ograniczeń w metodzie pracy potokowej. *Przegląd Naukowy Inżynieria i Kształtowanie Środowiska*, *26*(2), 210–218. https://doi.org/10.22630/PNIKS.2017.26.2.19

Marsh, E. R. (1975). The Harmonogram of Karol Adamiecki. *Academy of Management Journal*, *18*(2), 358–364. https://doi.org/10.2307/255537

Mendes Jr, R., Fernando, L., Heineck, M., & Vaca, O. L. (1998). New Applications of Line of Balance on Scheduling of Multi-Storey Buildings. *Association of Researchers in Construction Management*, *1*(December), 9–11.

Meng, T., & Pan, Q.-K. (2021). A distributed heterogeneous permutation flowshop scheduling problem with lot-streaming and carryover sequence-dependent setup time. *Swarm and Evolutionary Computation*, *60*. https://doi.org/10.1016/j.swevo.2020.100804

Mika, M., Waligóra, G., & Węglarz, J. (2005). Simulated annealing and tabu search for multi-mode resource-constrained project scheduling with positive discounted cash flows and different payment models. *European Journal of Operational Research*, *164*(3), 639–668. https://doi.org/10.1016/j.ejor.2003.10.053

Milat, M., Knezić, S., & Sedlar, J. (2021). Resilient Scheduling as a Response to Uncertainty in Construction Projects. *Applied Sciences*, *11*(14), 6493. https://doi.org/10.3390/app11146493

Milian, Z. (2006). *Metody określania rozkładu czasu realizacji przedsięwzięć budowlanych w acyklicznych sieciach stochastycznych*. Politechnika Krakowska. Seria: Inżynieria Lądowa.

Moselhi, O., & El-Rayes, K. (1993). Least cost scheduling for repetitive projects. *Canadian Journal of Civil Engineering*, *20*(5), 834–843. https://doi.org/10.1139/l93-109

Moselhi, O., & El-Rayes, K. (1993). Scheduling of Repetitive Projects with Cost Optimization. *Journal of Construction Engineering and Management*, *119*(4), 681–697. https://doi.org/10.1061/(ASCE)0733-9364(1993)119:4(681)

Mrozowicz, J. (1997). *Metody organizacji procesów budowlanych uwzględniające sprzężenia czasowe*. Dolnośląskie Wydawnictwo Edukacyjne.

O'Brien, J. J. (1975). VPM Scheduling for High-Rise Buildings. *Journal of the Construction Division*, *101*(4), 895–905.

Pempera, J., Smutnicki, C., & Wójcik, R. (2021). Minimizing the cycle time in the distributed flow shop scheduling problem. *IFAC-PapersOnLine*, *54*(1), 1081–1086. https://doi.org/10.1016/j.ifacol.2021.08.203

Pettie, S. (2002). *A Faster All-Pairs Shortest Path Algorithm for Real-Weighted Sparse Graphs* (s. 85–97). https://doi.org/10.1007/3-540-45465-9_9

Plebankiewicz, E. (2007). *Podstawy kosztorysowania robót budowlanych*. Wydawnictwo Politechniki Krakowskiej.

Podolski, M. (2008). *Analiza nowych zastosowań teorii szeregowania zadań w organizacji robót budowlanych*. Politechnika Wrocławska.

Podolski, M. (2016a). Scheduling of Job Resources in Multiunit Projects with the Use of Time I Cost Criteria. *Archives of Civil Engineering*, *62*(1), 143–158. https://doi.org/10.1515/ace-2015-0057

Podolski, M. (2016b). Management of resources in multiunit construction projects with the use of a tabu search algorithm. *Journal of Civil Engineering and Management*, *23*(2), 263–272. https://doi.org/10.3846/13923730.2015.1073616

Podolski, M., & Sroka, B. (2019). Cost optimization of Multiunit Construction Projects Using Linear Programming and Metaheuristic-based Simulated Annealing Algorithm. *JOURNAL OF CIVIL ENGINEERING AND MANAGEMENT*, *25*(8), 848–857. https://doi.org/10.3846/jcem.2019.11308

Radziszewska-Zielina, E., & Sroka, B. (2016). Metoda analityczna w podejściu probabilistycznym do planowania inwestycji wieloobiektowych. *MATERIAŁY BUDOWLANE*, *1*(6), 34–35, 38. https://doi.org/10.15199/33.2016.06.13

Reda, R. M. (1990). RPM: Repetitive Project Modeling. *Journal of Construction Engineering and Management*, *116*(2), 316–330. https://doi.org/10.1061/(ASCE)0733-9364(1990)116:2(316)

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, *21*(4), 25–34. https://doi.org/10.1145/37402.37406

Robert G. Sargent. (2010). Verification and validation of simulation models. *Proceedings of the 2010 Winter Simulation Conference*, *February*, 166–183. https://doi.org/10.1109/WSC.2010.5679166

Rogalska, M., Bozejko, W., & Hejducki, Z. (2008). Time/cost optimization using hybrid evolutionary algorithm in construction project scheduling. *Automation in Construction*. https://doi.org/10.1016/j.autcon.2008.04.002

Rogalska, M., Bożejko, W., Hejducki, Z., & Wodecki, M. (2009). Harmonogramowanie robót budowlanych z zastosowaniem algorytmu tabu search z rozmytymi czasami wykonania zadań. *Przegląd Budowlany*, *7–8*, 76–80.

Rogalska, M., & Hejducki, Z. (2004). Skracanie czasu realizacji przedsięwzięć budowlanych z zastosowaniem teorii ograniczeń TOC i metody CCS/BM. W *Bezpieczeństwo pożarowe, fizyka budowli, organizacja i zarządzanie w budownictwie, zagadnienia wybrane* (s. 197–204). Oficyna Wydawnicza Politechniki Warszawskiej.

Rogalska, M., & Hejducki, Z. (2007). Time buffers in construction process scheduling. *Journal of Civil Engineering and Management*, *13*(2), 143–148. https://doi.org/10.3846/13923730.2007.9636430

Rogalska, M., & Hejducki, Z. (2017). The application of time coupling methods in the engineering. *Czasopismo Techniczne*, *9*, 67–74. https://doi.org/10.4467/2353737XCT.17.148.7160

Rogalska, M., & Hejducki, Z. (2022). Zastosowanie metody sprzężeń czasowych TCM 1 w harmonogramowaniu robót budowlanych. *Materiały Budowlane*, *1*(12), 60–62. https://doi.org/10.15199/33.2022.12.15

Rosłon, J. H., & Kulejewski, J. E. (2019). A hybrid approach for solving multi-mode resource-constrained project scheduling problem in construction. *Open Engineering*, *9*(1), 7–13. https://doi.org/10.1515/eng-2019-0006

Rowiński, L. (1982). *Organizacja produkcji budowlanej*. Arkady.

Sabar, N. R., & Kendall, G. (2015). Population based Monte Carlo tree search hyper-heuristic for combinatorial optimization problems. *Information Sciences*, *314*, 225–239. https://doi.org/10.1016/j.ins.2014.10.045

San Cristóbal, J. R. (2009). Time, Cost, and Quality in a Road Building Project. *Journal of Construction Engineering and Management*, *135*(11), 1271–1274. https://doi.org/10.1061/(ASCE)CO.1943-7862.0000094

Schaefers, L., & Platzner, M. (2015). Distributed Monte Carlo Tree Search: A Novel Technique and its Application to Computer Go. *IEEE Transactions on Computational Intelligence and AI in Games*, *7*(4), 361–374. https://doi.org/10.1109/TCIAIG.2014.2346997

Senouci, A. B., & Eldin, N. N. (2004). Use of Genetic Algorithms in Resource Scheduling of Construction Projects. *Journal of Construction Engineering and Management*, *130*(6), 869–877. https://doi.org/10.1061/(ASCE)0733-9364(2004)130:6(869)

Smutnicki, C. (2012). *Algorytm szeregowania zadań*. Oficyna Wydawnicza Politechniki Wrocławskiej.

Sprawozdania finansowe za lata 2015-2021 Dom Development S.A. Pobrano 28 listopada 2022, z https://inwestor.domd.pl/pl/raporty-roczne

Sprawozdania Zarządu z działalności spółki Echo Investment S.A. za lata 2015-2021 . Pobrano 28 listopada 2022, z https://www.echo.com.pl/s,44,raporty-okresowe.html?year=0&cid=

Sroka, B., Rosłon, J., Podolski, M., Bożejko, W., Burduk, A., & Wodecki, M. (2021). Profit optimization for multi-mode repetitive construction project with cash flows using metaheuristics. *Archives of Civil and Mechanical Engineering*, *21*(2), 67. https://doi.org/10.1007/s43452-021-00218-2

Stradal, O., & Cacha, J. (1982). Time Space Scheduling Method. *Journal of the Construction Division*, *108*(CO3), 445–457.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, *64*(2), 278–285. https://doi.org/10.1016/0377-2217(93)90182-M

Thabet, W. Y., & Beliveau, Y. J. (1994). HVLS: Horizontal and Vertical Logic Scheduling for Multistory Projects. *Journal of Construction Engineering and Management*, *120*(4), 875–892. https://doi.org/10.1061/(ASCE)0733-9364(1994)120:4(875)

Tomar, A., & Bansal, V. K. (2019). Scheduling of repetitive construction projects using geographic information systems: an integration of critical path method and line of balance. *Asian Journal of Civil Engineering*, *20*(4), 549–562. https://doi.org/10.1007/s42107-019-00123-3

Tomczak, M. (2020). *Metoda harmonizacji wykonania procesów wieloobiektowego przedsięwzięcia budowlanego*. Politechnika Lubelska.

Tomczak, M., & Jaśkowski, P. (2022). Harmonizing construction processes in repetitive construction projects with multiple buildings. *Automation in Construction*, *139*, 104266. https://doi.org/10.1016/j.autcon.2022.104266

Tran, D.-H., Luong-Duc, L., Duong, M.-T., Le, T.-N., & Pham, A.-D. (2018). Opposition multiple objective symbiotic organisms search (OMOSOS) for time, cost, quality and work continuity tradeoff in repetitive projects. *Journal of Computational Design and Engineering*, *5*(2), 160–172. https://doi.org/10.1016/j.jcde.2017.11.008

Węglarz, J. (1981). *Sterowanie w systemach typu kompleks operacji*. Państwowe Wydawnictwo Naukowe.

Wyniki finansowe ATAL S.A. Pobrano 14 lipca 2022, z https://notowania.pb.pl/instrument/PLATAL000046/atal/raporty-finansowe/skonsolidowany/roczny/standardowy/1

Yew-Soon Ong, Meng-Hiot Lim, Ning Zhu, & Kok-Wai Wong. (2006). Classification of adaptive memetic algorithms: a comparative study. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, *36*(1), 141–152. https://doi.org/10.1109/TSMCB.2005.856143

Zhang, L., & Zou, X. (2015). Repetitive Project Scheduling: Theory and Methods. W *Repetitive Project Scheduling: Theory and Methods*. Elsevier. https://doi.org/10.1016/C2013-0-18950-7

Zhang, L., Zou, X., & Qi, J. (2015). A trade-off between time and cost in scheduling repetitive construction projects. *Journal of Industrial & Management Optimization*, *11*(4), 1423–1434. https://doi.org/10.3934/jimo.2015.11.1423

7. Załączniki

Załącznik 1: Kod funkcji realizującą czasowo-kosztowy model priorytetowego harmonogramowania

```
from pymprog import begin, var, maximize, solve, end, iprod, par, minimize, vobj, solver, glpk, sensitivity
from time import time
import numpy as np
from copy import copy
import xlsxwriter

class PS(object):

    def __init__(self, name, tgr, kgr=[], t_s=[], a_s=[], kindir=0, kp=[], kn=[], kc=[], Td=[],
                 Tso=[], Tfo=[], Tsb=[], Tfb=[], Ctol=[], Cwol=[], Ctou=[], Cwou=[], Ctbl=[], Cwbl=[], Ctbu=[], Cwbu=[], Ca=[],
                 ordinf=[1,1,1,1,0,1,1,0,0,1,1,0,0,0,0,0,0,1,1,1,1], pr=0, save=0, ret=0, ro=0.01):

        self.ro = ro

        self.name = name

        self.tgr = tgr
        self.kgr = kgr
        self.t_s = t_s
        self.a_s = a_s

        self.kindir = kindir

        self.kp = kp
        self.kn = kn
        self.kc = kc
        self.Td = Td

        self.Tso = Tso
        self.Tfo = Tfo
        self.Tsb = Tsb
        self.Tfb = Tfb

        self.Ctol= Ctol
        self.Cwol= Cwol
        self.Ctou= Ctou
        self.Cwou= Cwou

        self.Ctbl = Ctbl
        self.Cwbl = Cwbl
        self.Ctbu = Ctbu
        self.Cwbu = Cwbu

        self.Ca = Ca
        if self.Ca:
            self.Calen = len(self.Ca)
            self.Ctal = []
            self.Cwal = []
            self.Ctau = []
            self.Cwau = []
            self.S = []
            self.Sij = []
            self.P = []
            self.Pij = []
            for i in range(self.Calen):
                self.Ctal.append(self.Ca[i][0])
                self.Cwal.append(self.Ca[i][1])
                self.Ctau.append(self.Ca[i][2])
                self.Cwau.append(self.Ca[i][3])
                self.S.append(self.Ca[i][4])
                self.Sij.append([self.Ca[i][5], self.Ca[i][6]])
                self.P.append(self.Ca[i][7])
                self.Pij.append([self.Ca[i][8], self.Ca[i][9]])

        (self.n, self.m) = np.shape(self.tgr)

        self.ordinf = ordinf

        self.pr = pr

        self.ret = ret

        self.save = save

        self.vartab = [self.tgr, self.kgr, self.t_s, self.a_s, self.kindir, self.kp, self.kn, self.kc, self.Td, self.Tso, self.Tfo, self.Tsb, self.Tfb, self.Ctol,
        self.Cwol, self.Ctou, self.Cwou, self.Ctbl, self.Cwbl, self.Ctbu, self.Cwbu]
```

```
    self.sevebase()

def printdata(self):
    print("nazwa: ", self.name)
    print("tgr: ", self.tgr)
    print("kgr: ", self.kgr)
    print("t_s: ", self.t_s)
    print("a_s: ", self.a_s)
    print("kindir: ", self.kindir)
    print("kp: ", self.kp)
    print("kn: ", self.kn)
    print("kc: ", self.kc)
    print("Td: ", self.Td)
    print("Tso: ", self.Tso)
    print("Tfo: ", self.Tfo)
    print("Tsb: ", self.Tsb)
    print("Tfb: ", self.Tfb)
    print("Ctol: ", self.Ctol)
    print("Cwol: ", self.Cwol)
    print("Ctou: ", self.Ctou)
    print("Cwou: ", self.Cwou)
    print("Ctbl: ", self.Ctbl)
    print("Cwbl: ", self.Cwbl)
    print("Ctbu: ", self.Ctbu)
    print("Cwbu: ", self.Cwbu)
    print("Ca: ", self.Ca)

def validate(self):
    if not prod(
        [[len(self.t_s[i][j]) == len(self.a_s[i][j]) for i in range(self.n)] for j in range(self.m)]): return False
    if np.shape(self.kgr) != (self.n, self.m) or np.shape(self.t_s) != (self.n, self.m) or np.shape(self.a_s) != (
    self.n, self.m): return False
    if len(kp) != self.n or len(kn) != self.n or len(kc) != self.m: return False
    if len(self.Tso) != self.n or len(self.Tfo) != self.n or len(self.Tsb) != self.m or len(
        self.Tfb) != self.m: return False
    return True

def sevebase(self):
    self.bvartab=[]
    for i in self.vartab:
        self.bvartab.append(copy(i))
    self.bordinf = copy(self.ordinf)


def changeord(self,ord):
    ord = [x-1 for x in ord]
    for (i,j,k) in zip(self.bordinf,self.bvartab,self.vartab):
        if i and j and k:
            k.clear()
            for l in [j[m] for m in ord]:
                k.append(l)

def raport(self):
    print(self.name, ":", self.Time, " FC:", self.FC)

def solve(self,ord):

    self.changeord(ord)


    W = iprod(range(self.n), range(self.m))
    O = iprod(range(self.n - 1),
              range(self.m))
    B = iprod(range(self.n),
              range(self.m - 1))


    if self.t_s:
        A = [(i, j, k) for i in range(self.n) for j in range(self.m) for k in
             range(len(self.t_s[i][j]))]
    else:
        A = []

    if self.Ca:
        AA = range(self.Calen)

    start_time = time()

    PrSc = begin('TCM')
```

```
tgr = par('tgr', self.tgr)
kgr = par('kgr', self.kgr)
t_s = par('t_s', self.t_s)
a_s = par('a_s', self.a_s)

kindir = par('kindir', self.kindir)

kp = par('kp', self.kp)
kn = par('kn', self.kn)
kc = par('kc', self.kc)
Td = par('Td', self.Td)

Tso = par('Tso', self.Tso)
Tfo = par('Tfo', self.Tfo)
Tsb = par('Tsb', self.Tsb)
Tfb = par('Tfb', self.Tfb)

Ctol = par('Ctol', self.Ctol)
Cwol = par('Cwol', self.Cwol)
Ctou = par('Ctou', self.Ctou)
Cwou = par('Cwou', self.Cwou)
Ctbl = par('Ctbl', self.Ctbl)
Cwbl = par('Cwbl', self.Cwbl)
Ctbu = par('Ctbu', self.Ctbu)
Cwbu = par('Cwbu', self.Cwbu)

if self.Ca:
   Ctal = par('Ctal',self.Ctal)
   Cwal = par('Cwal',self.Cwal)
   Ctau = par('Ctau',self.Ctau)
   Cwau = par('Cwau',self.Cwau)

t = var('t', W)
d_s = var('d_s', A)
NWR = var('NWR', W)
NWZ = var('NWZ', W)
NPR = var('NPR', W)
NPZ = var('NPZ', W)
ZC = var('ZC', W)
Ckol = var('Ckol', O)
Ckou = var('Ckou', O)
Ckbl = var('Ckbl', B)
Ckbu = var('Ckbu', B)

if self.Ca:
   Ckal = var('Ckal', AA)
   Ckau = var('Ckau', AA)

p = var("p", range(self.n))
n = var("n", range(self.n))


Kdir = sum(
   [sum([d_s[i, j, s] * a_s[i][j][s] for s in range(len(self.t_s[i][j]))]) + kgr[i][j] for (i, j) in W])
Kindir = NWZ[self.n - 1, self.m - 1] * kindir
Kp = sum([kp[i] * p[i] for i in range(self.n)])
Kn = sum([kn[i] * n[i] for i in range(self.n)])
Kc = sum(
   [(NWZ[self.n - 1, j] - NWR[0, j] - sum([t[i, j] for i in range(self.n)])) * kc[j] for j in range(self.m)])
KCo = sum([Cwol[i][j] * Ckol[i, j] + Cwou[i][j] * Ckou[i, j] for (i, j) in O])
KCb = sum([Cwbl[i][j] * Ckbl[i, j] + Cwbu[i][j] * Ckbu[i, j] for (i, j) in B])

if self.Ca:
   KCa=sum([Cwal[i]*Ckal[i]+Cwau[i]*Ckau[i] for i in AA])
else:
   KCa = 0
KT = sum(NWZ[i, j] - NPZ[i, j] for (i, j) in W) * self.ro

minimize(Kdir + Kindir + Kp - Kn + Kc + KCo + KCb + KCa + KT)

for (i,j) in W:
   tgr[i][j] <= t[i,j] <= tgr[i][j] + sum([ts for ts in t_s[i][j]])
   for k in range(len(t_s[i][j])):
      d_s[i,j,k] <= t_s[i][j][k]
      tgr[i][j] + sum([d_s[i,j,l] for l in range(k+1)]) <= t[i,j]

for (i, j) in W:
   NWZ[i, j] == NWR[i, j] + t[i, j]
   NPZ[i, j] == NPR[i, j] + t[i, j]
   ZC[i, j] == NPZ[i, j] - NWZ[i, j]
```

```
for (i, j) in O:
   NWR[i + 1, j] >= NWZ[i, j] + Ctol[i][j] - Ckol[i, j]
   NWR[i + 1, j] <= NWZ[i, j] + Ctou[i][j] + Ckou[i, j]

   NPR[i + 1, j] >= NPZ[i, j] + Ctol[i][j] - Ckol[i, j]
   NPR[i + 1, j] <= NPZ[i, j] + Ctou[i][j] + Ckou[i, j]

for (i, j) in B:
   NWR[i, j + 1] >= NWZ[i, j] + Ctbl[i][j] - Ckbl[i, j]
   NWR[i, j + 1] <= NWZ[i, j] + Ctbu[i][j] + Ckbu[i, j]

   NPR[i, j + 1] >= NPZ[i, j] + Ctbl[i][j] - Ckbl[i, j]
   NPR[i, j + 1] <= NPZ[i, j] + Ctbu[i][j] + Ckbu[i, j]

for i in range(self.n):
   NWZ[i, self.m - 1] - p[i] + n[i] == Td[i]

   NWR[i, 0] - Tso[i] >= 0
   NPZ[i, self.m - 1] - Tfo[i] <= 0

for j in range(self.m):
   NWR[0, j] - Tsb[j] >= 0
   NPZ[self.n - 1, j] - Tfb[j] <= 0


if self.Ca:
   for i in range(self.Calen):
      if self.S[i] == "NWR":
         S=NWR[self.Sij[i][0],self.Sij[i][1]]
      elif self.S[i] == "NWZ":
         S=NWZ[self.Sij[i][0],self.Sij[i][1]]
      elif self.S[i] == "NPR":
         S=NPR[self.Sij[i][0],self.Sij[i][1]]
      elif self.S[i] == "NPZ":
         S=NPZ[self.Sij[i][0],self.Sij[i][1]]
      elif self.S[i] == "ZC":
         S=ZC[self.Sij[i][0],self.Sij[i][1]]

      if self.P[i] == "NWR":
         P=NWR[self.Pij[i][0],self.Pij[i][1]]
      elif self.P[i] == "NWZ":
         P=NWZ[self.Pij[i][0],self.Pij[i][1]]
      elif self.P[i] == "NPR":
         P=NPR[self.Pij[i][0],self.Pij[i][1]]
      elif self.P[i] == "NPZ":
         P=NPZ[self.Pij[i][0],self.Pij[i][1]]
      elif self.P[i] == "ZC":
         P=ZC[self.Pij[i][0],self.Pij[i][1]]

      S >= P + Ctal[i] - Ckal[i]
      S <= P + Ctau[i] + Ckau[i]


NWZ[self.n - 1, self.m - 1] == NPZ[self.n - 1, self.m - 1]

solver('simplex', msg_lev=glpk.GLP_MSG_OFF)

PrSc.solve()

KCop = sum([Cwol[i][j].value * Ckol[i, j].primal + Cwou[i][j].value * Ckou[i, j].primal for (i, j) in O])
KCbp = sum([Cwbl[i][j].value * Ckbl[i, j].primal + Cwbu[i][j].value * Ckbu[i, j].primal for (i, j) in B])

if self.Ca:
   KCap = sum([Cwal[i].value * Ckal[i].primal + Cwau[i].value * Ckau[i].primal for i in AA])
else:
   KCap = 0

KTp = sum(NWZ[i, j].primal - NPZ[i, j].primal for (i, j) in W) * self.ro

costs = vobj() - (KCop + KCbp + KCap + KTp)

if self.pr:
   print('Nazwa: ', self.name)
   print('FC:',vobj())
   for (i,j) in W:
      print('O',i+1,'
      B',j+1,'NWR',NWR[i,j].primal,'NWZ',NWZ[i,j].primal,'NPR',NPR[i,j].primal,'NPZ',NPZ[i,j].primal,'ZC',ZC[i,j].primal)
   for (i,j) in O:
      print(i+1,j+1,'Ckol',Ckol[i,j].primal,'Ckou',Ckou[i,j].primal)
   for (i,j) in B:
      print(i+1,j+1,'Ckbl',Ckbl[i,j].primal,'Ckbu',Ckbu[i,j].primal)
```

```
        if self.Ca:
            for i in range(self.Calen):
                print(f'Si: {self.Ca[i][5]} Sj: {self.Ca[i][6]} Pi: {self.Ca[i][8]} Pj: {self.Ca[i][9]} S: {self.Ca[i][4]} P: {self.Ca[i][7]} Ckal: {Ckal[i].primal} Ckau: {Ckau[i].primal}')
        for i in range(self.n):
            print('p',i+1,p[i].primal)
        print('Termin realizacji:',NWZ[self.n-1,self.m-1].primal)

        print(f'Koszt realizacji:{costs}')

        print('Czas kompilacji:',round(time()-start_time,4),'s')

    if self.save:
        workbook = xlsxwriter.Workbook(self.name+'.xlsx')
        worksheet = workbook.add_worksheet()
        merge_format = workbook.add_format({'align': 'center'})
        for i in range(self.n):
            for j in range(self.m):
                worksheet.write(i*4, j*4, f'NWR:{NWR[i,j].primal:.0f}',merge_format)
                worksheet.write(i*4, j*4+1, f't:{NWZ[i, j].primal-NWR[i,j].primal:.0f}',merge_format)
                worksheet.write(i*4, j*4+2, f'NWZ:{NWZ[i, j].primal:.0f}',merge_format)
                worksheet.write(i*4+1,j*4+1, f'O{i+1} B{j+1}',merge_format)
                worksheet.write(i*4+2, j*4, f'NPR:{NPR[i, j].primal:.0f}',merge_format)
                worksheet.write(i*4+2, j*4+1, f'ZC:{ZC[i, j].primal:.0f}',merge_format)
                worksheet.write(i*4+2, j*4+2, f'NPZ:{NPZ[i, j].primal:.0f}',merge_format)
        workbook.close()

    if self.ret==0:
        return vobj()
    elif self.ret==1:
        NWRr = np.zeros((self.n,self.m))
        NWZr = np.zeros((self.n,self.m))
        NPRr = np.zeros((self.n,self.m))
        NPZr = np.zeros((self.n,self.m))
        for i in range(self.n):
            for j in range(self.m):
                NWRr[i,j] = NWR[i,j].primal
                NWZr[i,j] = NWZ[i,j].primal
                NPRr[i,j] = NPR[i,j].primal
                NPZr[i,j] = NPZ[i,j].primal

        return vobj(), NWRr, NWZr, NPRr, NPZr, costs

def getNumber(self):
    return self.n
```

Załącznik 2: Kod dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od A.1-A.8

```
import re
import random
import math
from time import time, sleep
from numpy import flip, array
from itertools import permutations

class MCTS(object):

    def __init__(self, calc_engine, c, constrains=None, brutemaxnumber=8, exectime=30, mintime=1,
                 genVisualization=False, selectbestnodemode=2, selectUCBmode=3, log=1, gtt=1, dLM=1):
        self.CalcEngine = calc_engine
        self.C = c

        if constrains is None:
            constrains = []

        self.brutemaxnumber = brutemaxnumber
        self.exectime = exectime
        self.mintime = mintime
        self.genVisualization = genVisualization
        self.selectbestnodemode = selectbestnodemode
        self.selectUCBmode = selectUCBmode
        self.log = log
        self.gtt = gtt

        if dLM == 1:
            self.defLM = defLegMove

        elif dLM == 2:
            self.defLM = defLegMoveSJ1
        elif dLM == 3:
            self.defLM = defLegMoveSJ2
        elif dLM == 4:
            self.defLM = defLegMoveSJ3

        if self.log:
            self.f = open("data/log2.txt", 'a')
            self.f.write("*" * 100 + "\n")
            self.f.write(
                "C=%s; Brutemaxnumber= %s; Exectime=%s; mintime=%s; selectbestnodemode=%s; selectUCBmode=%s\n" % (
                    self.C, self.brutemaxnumber, self.exectime, self.mintime, self.selectbestnodemode,
                    self.selectUCBmode))

        self.simiter = 0
        self.bestval = math.pow(10, 100)
        self.bestord = []
        self.number = self.CalcEngine.getNumber()
        self.con = recon(constrains)
        self.actnumber = self.number

        if self.actnumber > self.brutemaxnumber:
            self.root = node(self, list(range(1, self.number + 1)), [], 1, None)
            self.act = []
            self.bestnode = self.root
            self.starttime = time()
            self.acttime = self.starttime
            self.timetable = self.gentimetable()

            precent = 0

            while self.actnumber > self.brutemaxnumber:
                i = 0
                while time() - self.acttime < self.timetable[self.actnumber - 1]:
                    self.act = self.root
                    self.selection()
                    self.expansion()
                    (r, o) = self.simulation()

                    self.savebest(r, o, self.number - self.actnumber)
                    self.backpropagation(r)
                    i += 1
                    t = ((time() - self.starttime) / self.exectime)
                    if t >= precent:
                        print('{0:.0%}'.format(precent))
                        precent += 0.05

                if self.genVisualization:
                    self.genGraphVisualization()
```

```
            self.bestnode = self.selectbestnode(self.bestnode, self.selectbestnodemode)
            self.removesiblings(self.bestnode)

            self.actnumber -= 1
            self.acttime = time()

        self.bruteforce()

    if self.log:
        self.f.close()

def selection(self):
    n = self.root
    while (not n.move) and (n.successor):
        bestval = -pow(100, 10)
        for i in n.successor:
            i.UCB = i.UCBc(self.selectUCBmode)
            if i.UCB > bestval:
                best = i
                bestval = i.UCB
        n = best
    self.act = n

def expansion(self):
    if self.act.move:
        m = random.choice(self.act.move)
        self.act.move.remove(m)

        n = self.act.actmove[:]
        n.remove(m)

        k = self.act.premove[:]
        k.append(m)

        newnode = node(self, n, k, self.act.movecount + 1, self.act)
        self.act.successor.append(newnode)
        self.act = newnode

def simulation(self):

    if not self.act.actmove:
        return [self.fitfunc(self.act.premove), self.act.premove]

    if not self.act.legMove:
        return 0, 0

    f = False
    i = 0

    while (not f) and (i < 1000):
        actmove = list(self.act.actmove)
        premove = list(self.act.premove)
        movecount = self.act.movecount
        legMove = self.defLM(self.con, actmove, movecount, premove)
        while actmove and legMove:
            m = random.choice(legMove)
            premove.append(m)
            actmove.remove(m)
            movecount += 1
            if not actmove:
                f = True
            else:
                legMove = self.defLM(self.con, actmove, movecount, premove)
        i = i + 1
    if f:
        self.simiter += 1
        return [self.fitfunc(premove), premove]
    else:
        return 0, 0

def backpropagation(self, r):

    while self.act.predecessor:
        self.act.si += 1
        self.act.predecessor.spsi += 1
        if r:
            self.act.wi += r
            self.act.ssi += 1
            self.act.predecessor.spssi += 1
            if r < self.act.nbestval:
                self.act.nbestval = r
```

```
            self.act = self.act.predecessor
        if r < self.root.nbestval:
            self.root.best = r
        self.root.si += 1
        if r:
            self.root.wi += r
            self.root.ssi += 1

    def genGraphVisualization(self):
        params = tlp.getDefaultPluginParameters("Hierarchical Graph", self.graph)
        params["orientation"] = "vertical"
        params["layer spacing"] = 1.0
        params["node spacing"] = 1.0
        viewLayout = self.graph.getLayoutProperty("viewLayout")
        self.graph.getStringProperty("viewLabel")
        self.graph.applyLayoutAlgorithm("Hierarchical Graph", viewLayout, params)
        tlp.saveGraph(self.graph, 'g%s.tlp' % self.actnumber)

    def fitfunc(self, ord):
        return self.CalcEngine.solve(ord)

    def savebest(self, r, o, m):

        if r and r < self.bestval:
            self.bestval = r
            self.bestord = o
            if self.log:
                self.f.write("{};{};{};{};{}\n".format(time() - self.starttime, self.bestval, m, self.simiter,
                                                       self.bestord))
            print(r, o)

    def selectbestnode(self, n, opt):
        if opt == 1:
            bestval = -pow(10, 10)
            for i in n.successor:
                i.UCB = i.UCBc(self.selectUCBmode)
                if i.UCB > bestval:
                    best = i
                    bestval = i.UCB
            return best
        elif opt == 2:
            bestval = pow(10, 10)
            for i in n.successor:
                if i.nbestval < bestval:
                    best = i
                    bestval = i.nbestval
            return best
        elif opt == 3:
            bestval = pow(10, 10)
            for i in n.successor:
                if i.si > bestval:
                    best = i
                    bestval = i.si
            return best

    def removesiblings(self, bestnode):
        for i in bestnode.predecessor.successor:
            if i != bestnode:
                self.removesubtree(i)
        bestnode.predecessor.successor = [bestnode]

    def removesubtree(self, t):
        for i in t.successor:
            self.removesubtree(i)
            i.successor = []

    def gentimetable(self):
        if self.gtt == 1:
            timeparam = 0.9
            t = array([pow(timeparam, i) for i in range(1, self.number + 1)])
            pt = self.exectime / sum(t)
            t = flip(t * pt, 0)

        elif self.gtt == 2:
            t = array([0 for i in range(1, self.number + 1)])
            t[0] = self.exectime
            t = flip(t, 0)
        t[t < self.mintime] = self.mintime
        return t

    def bruteforce(self):
```

```
        self.bruteforcestarttime = time()
        it = 0
        maxit = len(list(permutations(self.bestnode.actmove)))
        for i in permutations(self.bestnode.actmove):
            if it % 1000 == 0:
                print(it / maxit)
            it += 1
            ord = self.bestnode.premove + list(i)
            if self.isLegalMove(ord):
                self.savebest(self.fitfunc(ord), ord, "Bruteforce")

    def isLegalMove(self, ord):
        for i in range(len(ord)):
            f = self.defLM(self.con, [ord[i]], i, ord[:i])
            if not f:
                break
        return f


class node(object):
    def __init__(self, master, actmove, premove, movecount, predecessor):
        self.master = master
        self.actmove = actmove
        self.premove = premove
        self.movecount = movecount
        self.predecessor = predecessor
        self.legMove = master.defLM(self.master.con, self.actmove, self.movecount,
                                    self.premove)
        self.move = self.legMove[:]
        self.successor = []
        self.nbestval = pow(10, 10)
        self.wi = 0
        self.si = 0
        self.ssi = 0
        self.spsi = 0
        self.spssi = 0
        self.UCB = 0

    def UCBc(self, mode):
        if mode == 1:
            if self.ssi == 0:
                return (self.master.C) * math.sqrt((math.log(self.predecessor.spsi) / self.si))
            else:
                return -(self.wi / self.ssi) / self.master.bestval + (self.master.C) * math.sqrt(
                    (math.log(self.predecessor.spsi) / self.si))
        elif mode == 2:
            if self.ssi == 0:
                return (self.master.C) * math.sqrt((math.log(self.master.bestnode.spsi) / self.si))
            else:
                return -(self.wi / self.ssi) / self.master.bestval + (self.master.C) * math.sqrt(
                    (math.log(self.master.bestnode.spsi) / self.si))
        elif mode == 3:
            if self.ssi == 0:
                return (self.master.C) * math.sqrt((math.log(self.master.root.spsi) / self.si))
            else:
                return -(self.wi / self.ssi) / self.master.bestval + (self.master.C) * math.sqrt(
                    (math.log(self.master.root.spsi) / self.si))
        elif mode == 4:
            return -self.nbestval / self.master.bestval + (self.master.C) * math.sqrt(
                (math.log(self.predecessor.spsi) / self.si))
        elif mode == 5:
            return -self.nbestval / self.master.bestval + (self.master.C) * math.sqrt(
                (math.log(self.master.bestnode.spsi) / self.si))
        elif mode == 6:
            return -self.nbestval / self.master.bestval + (self.master.C) * math.sqrt(
                (math.log(self.master.root.spsi) / self.si))

    def pr(self):
        print("*" * 50)
        print("actmove: %s" % self.actmove)
        print("premove: %s" % self.premove)
        print("predecessor: %s" % self.predecessor)
        print("legMove: %s" % self.legMove)
        print("move: %s" % self.move)
        print("successor: %s" % self.successor)
        print("movecount %s" % self.movecount)
        print("wi: %s" % self.wi)
        print("si: %s" % self.si)
        print("UCB: %s" % self.UCB)
```

```
def recon(constrains):
    con1 = []
    con2 = []
    con3 = []
    con4 = []
    con5 = []
    for i in constrains:
        if re.match("[0-9]+=[0-9]+", i):
            con1.append(re.split("=", i))
        elif re.match("[0-9]+>>[0-9]+", i):
            con2.append(re.split(">>", i))
        elif re.match("[0-9]+>[0-9]+", i):
            con3.append(re.split(">", i))
        elif re.match("[0-9]+<>[0-9]+", i):
            con4.append(re.split("<>", i))
        elif re.match("[0-9]+_[0-9]+", i):
            con5.append(re.split("_", i))
        else:
            print("Złe ograniczenie")
    con1 = [list(map(int, i)) for i in con1]
    con2 = [list(map(int, i)) for i in con2]
    con3 = [list(map(int, i)) for i in con3]
    con4 = [list(map(int, i)) for i in con4]
    con5 = [list(map(int, i)) for i in con5]
    return [con1, con2, con3, con4, con5]


def defLegMove(con, actmove, movecount, premove):
    lm = []

    for i in actmove:
        legal = True
        for j in con[0]:
            if (j[1] != movecount) == (j[0] == i):
                legal = False

        if premove:
            for j in con[1]:
                if (j[0] != premove[-1]) == (j[1] == i):
                    legal = False
        else:
            for j in con[1]:
                if j[1] == i:
                    legal = False

        if premove:
            for j in con[2]:
                if (j[0] not in premove) and (j[1] == i):
                    legal = False
        else:
            for j in con[2]:
                if j[1] == i:
                    legal = False

        if premove:
            for j in con[3]:
                if (j[0] == premove[-1] and j[1] != i and j[1] not in premove) or (
                        j[1] == premove[-1] and j[0] != i and j[0] not in premove):
                    legal = False

        if premove:
            for j in con[4]:
                if not set(j).issubset(premove):
                    if (premove[-1] in j) and i not in j:
                        legal = False
        if legal:
            lm.append(i)
    return (lm)


def defLegMoveSJ1(con, actmove, movecount, premove):
    lm = []

    O110 = range(1, 21)
    O120 = range(21, 31)
    O150 = range(31, 41)

    O = [O110, O120, O150]

    for i in O:
        b = sorted(list(set(set(i) - set(premove))))
```

```
        if b:
            lm.append(b[0])

    return (lm)


def defLegMoveSJ2(con, actmove, movecount, premove):
    lm = []

    O110 = range(1, 21)
    O120 = range(21, 31)
    O150 = range(31, 41)

    B = [O120, O150]

    O = [O110, O120, O150]

    for i in B:
        if len(list(set(set(i) - set(premove)))) % 2 == 1:
            b = sorted(list(set(set(i) - set(premove))))
            return [b[0]]

    for i in O:
        b = sorted(list(set(set(i) - set(premove))))
        if b:
            lm.append(b[0])
    return (lm)


def defLegMoveSJ3(con, actmove, movecount, premove):
    lm = []

    O110 = range(1, 21)
    O120 = range(21, 31)
    O150 = range(31, 41)

    B = [O110, O120, O150]

    O = [O110, O120, O150]

    for i in B:
        if len(list(set(set(i) - set(premove)))) % 2 == 1:
            b = sorted(list(set(set(i) - set(premove))))
            return [b[0]]

    for i in O:
        b = sorted(list(set(set(i) - set(premove))))
        if b:
            lm.append(b[0])
    return (lm)


if __name__ == "__main__":
    print(recon(['3=2', '4>>5', '8>9', '6<>7', "6_7_8", "1_2_3"]))
```

Załącznik 3: Kod dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od A.1-A.8

Plik PSa1-8.py

```
from PS import PS

'''Stałe'''
L1 = 1e4
L2 = 1e6
L3 = 1e8
L4 = 1e10
L5 = 1e12
s=2

'''Typ efektu A1'''
name = "A1"
tgr = [[7, 8, 6, 7],
       [9, 4, 7, 9],
       [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
        [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
        [0, 0, 0],
        [0, 0, 0]]
Cwbl = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])


'''Typ efektu A2'''
name = "A2"
tgr = [[7, 8, 6, 7],
       [9, 4, 7, 9],
       [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
```

```
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol =  [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwou =  [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu A3'''

```
name = "A3"
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
   [[0], [0], [0], [0]],
   [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
   [[0], [0], [0], [0]],
   [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
```

```
'''Typ efektu A4'''
name = "A4"
tgr = [[7, 8, 6, 7],
      [9, 4, 7, 9],
      [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
      [0, 0, 0, 0],
      [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
       [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
       [L1, L1, L1, L1]]
Ctou = [[L1, L1,  0, L1],
       [L1, L1,  0, L1]]
Cwou = [[L1, L1, L1, L1],
       [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
       [0, 0, 0],
       [0, 0, 0]]
Cwbl = [[L1, L1, L1],
       [L2, L2, L2],
       [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
       [0, 0, 0],
       [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
       [L2, L2, L2],
       [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu,
     Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
'''Typ efektu A5'''
name = "A5"
tgr = [[7, 8, 6, 7],
      [9, 4, 7, 9],
      [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
      [0, 0, 0, 0],
      [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[-s, -s, -s, -s],
       [-s, -s, -s, -s]]
Cwol = [[L1, L1, L1, L1],
       [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
       [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
       [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
```

```
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu A6'''

```
name = "A6"
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[-s, -s, -s],
      [-s, -s, -s],
      [-s, -s, -s]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu A7'''

```
name = "A7"
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
```

```
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[-s, -s, -s, -s],
      [-s, -s, -s, -s]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[-s, -s, -s],
      [-s, -s, -s],
      [-s, -s, -s]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu A8a'''

```
name = "A8a"
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L2, L4, L1],
      [L1, L2, L4, L1]]
Ctou = [[L1,  0,  0, L1],
      [L1,  0,  0, L1]]
Cwou = [[L1, L2, L4, L1],
      [L1, L2, L4, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L3, L3, L3],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [ 0,  0,  0],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L3, L3, L3],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
```

```
t.solve([1, 2, 3])

'''Typ efektu A8b'''
name = "A8b"
tgr = [[7, 8, 6, 7],
      [9, 4, 7, 9],
      [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
      [0, 0, 0, 0],
      [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L5, L5, L4, L5],
      [L5, L5, L4, L5]]
Ctou = [[L1,  0,  0, L1],
      [L1,  0,  0, L1]]
Cwou = [[L1, L2, L4, L1],
      [L1, L2, L4, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L5, L5, L5],
      [L3, L3, L3],
      [L5, L5, L5]]
Ctbu = [[L1, L1, L1],
      [ 0,  0,  0],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L3, L3, L3],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = [[], [], [], []]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, pr=1, save=1)
t.solve([1, 2, 3])
```

```
from PS import PS

'''Stałe'''
L1 = 1e4
L2 = 1e6
L3 = 1e8
L4 = 1e10
L5 = 1e12
L6 = 1e14
s=2

'''Typ efektu B1'''

name = "B1"
n=3
m=4
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
   [[0], [0], [0], [0]],
   [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
   [[0], [0], [0], [0]],
   [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
     [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
     [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
     [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
     [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
     [0, 0, 0],
     [0, 0, 0]]
Cwbl = [[L1, L1, L1],
     [L1, L1, L1],
     [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
     [L1, L1, L1],
     [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
     [L1, L1, L1],
     [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = []

for i in range(n):
   for j in range(m):
      if j > 0 and i < n - 1:
         Ca.append([0, L1, L1, L1, "NWR", i, j, "NWR", i + 1, j - 1])
         Ca.append([0, L1, L1, L1, "NPR", i, j, "NPR", i + 1, j - 1])

t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])

'''Typ efektu B2'''

name = "B2"
n=3
m=4
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
```

```
      [0, 0, 0, 0],
      [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = []

for i in range(n):
   for j in range(m):
      if j < m - 1 and i > 0:
         Ca.append([0, L1, L1, L1, "NWR", i, j, "NWR", i - 1, j + 1])
         Ca.append([0, L1, L1, L1, "NPR", i, j, "NPR", i - 1, j + 1])

t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu B3'''

```
name = "B3"
n=3
m=4
tgr = [[7, 8, 6, 7],
      [9, 4, 7, 9],
      [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
      [0, 0, 0, 0],
      [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
```

```
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = []

for i in range(n):
   for j in range(m):
      if j > 0 and i < n - 1:
         Ca.append([0, L2, 0, L2, "NWR", i, j, "NWR", i + 1, j - 1])

t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu B4'''

```
name = "B4"
n=3
m=4
k=2
tgr = [[7, 8, 6, 7],
      [9, 4, 7, 9],
      [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
      [0, 0, 0, 0],
      [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca = []

for i in range(n-1):
   Ca.append([1, L2, L2, L2, "ZC", i+1, k, "ZC", i, k])
```

```
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])
```

'''Typ efektu B5'''

```
name = "B5"
n=3
m=4
k=2
tgr = [[7, 8, 6, 7],
       [9, 4, 7, 9],
       [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
kindir = 10
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
        [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
        [0, 0, 0],
        [0, 0, 0]]
Cwbl = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]

for i in range(n):
    for j in range(m):
        if j > 0 and i < n - 1:
            Ca.append([0, L2, 0, L2, "NWR", i, j, "NWZ", i + 1, j - 1])

t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])
```

```
import re
import random
import math
from time import time, sleep
from numpy import flip, array
from itertools import permutations


class MCTS(object):
    def __init__(self, calc_engine, c, constrains=None, brutemaxnumber=8, exectime=30, mintime=1,
                 genVisualization=False, selectbestnodemode=2, selectUCBmode=3, log=1, gtt=1, dLM=1):

        self.CalcEngine = calc_engine

        self.C = c

        if constrains is None:
            constrains = []

        self.brutemaxnumber = brutemaxnumber
        self.exectime = exectime
        self.mintime = mintime
        self.genVisualization = genVisualization
        self.selectbestnodemode = selectbestnodemode
        self.selectUCBmode = selectUCBmode
        self.log = log

        self.gtt = gtt

        if dLM == 1:
            self.defLM = defLegMove

        elif dLM == 2:
            self.defLM = defLegMoveSJ1
        elif dLM == 3:
            self.defLM = defLegMoveSJ2
        elif dLM == 4:
            self.defLM = defLegMoveSJ3

        if self.log:
            self.f = open("data/log2.txt", 'a')
            self.f.write("*" * 100 + "\n")
            self.f.write(
                "C=%s; Brutemaxnumber= %s; Exectime=%s; mintime=%s; selectbestnodemode=%s; selectUCBmode=%s\n" % (
                    self.C, self.brutemaxnumber, self.exectime, self.mintime, self.selectbestnodemode,
                    self.selectUCBmode))

        self.simiter = 0

        self.bestval = math.pow(10, 100)
        self.bestord = []

        self.number = self.CalcEngine.getNumber()
        self.con = recon(constrains)
        self.actnumber = self.number

        if self.actnumber > self.brutemaxnumber:
            self.root = node(self, list(range(1, self.number + 1)), [], 1, None)
            self.act = []
            self.bestnode = self.root
            self.starttime = time()
            self.acttime = self.starttime
            self.timetable = self.gentimetable()

            precent = 0

            while self.actnumber > self.brutemaxnumber:
                i = 0
                while time() - self.acttime < self.timetable[self.actnumber - 1]:
                    self.act = self.root
                    self.selection()
                    self.expansion()
                    (r, o) = self.simulation()

                    self.savebest(r, o, self.number - self.actnumber)
                    self.backpropagation(r)
                    i += 1
                    t = ((time() - self.starttime) / self.exectime)
                    if t >= precent:
```

```
                print('{0:.0%}'.format(precent))
                precent += 0.05

            if self.genVisualization:
                self.genGraphVisualization()

            self.bestnode = self.selectbestnode(self.bestnode, self.selectbestnodemode)

            self.removesiblings(self.bestnode)

            self.actnumber -= 1
            self.acttime = time()

        self.bruteforce()
    else:
        pass

    if self.log:
        self.f.close()

def selection(self):
    n = self.root
    while (not n.move) and (n.successor):
        bestval = -pow(100, 10)
        for i in n.successor:
            i.UCB = i.UCBc(self.selectUCBmode)
            if i.UCB > bestval:
                best = i
                bestval = i.UCB
        n = best
    self.act = n

def expansion(self):
    if self.act.move:
        m = random.choice(self.act.move)
        self.act.move.remove(m)

        n = self.act.actmove[:]
        n.remove(m)

        k = self.act.premove[:]
        k.append(m)

        newnode = node(self, n, k, self.act.movecount + 1, self.act)
        self.act.successor.append(newnode)
        self.act = newnode

def simulation(self):
    if not self.act.actmove:
        return [self.fitfunc(self.act.premove), self.act.premove]

    if not self.act.legMove:
        return 0, 0

    f = False
    i = 0

    while (not f) and (i < 1000):
        actmove = list(self.act.actmove)
        premove = list(self.act.premove)
        movecount = self.act.movecount
        legMove = self.defLM(self.con, actmove, movecount, premove)
        while actmove and legMove:
            m = random.choice(legMove)
            premove.append(m)
            actmove.remove(m)
            movecount += 1
            if not actmove:
                f = True
            else:
                legMove = self.defLM(self.con, actmove, movecount, premove)
        i = i + 1
    if f:
        self.simiter += 1
        return [self.fitfunc(premove), premove]
    else:
        return 0, 0

def backpropagation(self, r):
    while self.act.predecessor:
        self.act.si += 1
```

```
            self.act.predecessor.spsi += 1
            if r:
                self.act.wi += r
                self.act.ssi += 1
                self.act.predecessor.spssi += 1
                if r < self.act.nbestval:
                    self.act.nbestval = r
            self.act = self.act.predecessor
        if r < self.root.nbestval:
            self.root.best = r
        self.root.si += 1
        if r:
            self.root.wi += r
            self.root.ssi += 1

    def fitfunc(self, ord):
        return self.CalcEngine.solve(ord)

    def savebest(self, r, o, m):

        if r and r < self.bestval:
            self.bestval = r
            self.bestord = o
            if self.log:
                self.f.write("{};{};{};{};{}\n".format(time() - self.starttime, self.bestval, m, self.simiter,
                                                       self.bestord))
            print(r, o)

    def selectbestnode(self, n, opt):
        if opt == 1:
            bestval = -pow(10, 10)
            for i in n.successor:
                i.UCB = i.UCBc(self.selectUCBmode)
                if i.UCB > bestval:
                    best = i
                    bestval = i.UCB
            return best
        elif opt == 2:
            bestval = pow(10, 10)
            for i in n.successor:
                if i.nbestval < bestval:
                    best = i
                    bestval = i.nbestval
            return best
        elif opt == 3:
            bestval = pow(10, 10)
            for i in n.successor:
                if i.si > bestval:
                    best = i
                    bestval = i.si
            return best

    def removesiblings(self, bestnode):
        for i in bestnode.predecessor.successor:
            if i != bestnode:
                self.removesubtree(i)
        bestnode.predecessor.successor = [bestnode]

    def removesubtree(self, t):
        for i in t.successor:
            self.removesubtree(i)
            i.successor = []

    def gentimetable(self):
        if self.gtt == 1:
            timeparam = 0.9
            t = array([pow(timeparam, i) for i in range(1, self.number + 1)])
            pt = self.exectime / sum(t)
            t = flip(t * pt, 0)

        elif self.gtt == 2:
            t = array([0 for i in range(1, self.number + 1)])
            t[0] = self.exectime
            t = flip(t, 0)
        t[t < self.mintime] = self.mintime
        return t

    def bruteforce(self):
        self.bruteforcestarttime = time()
        it = 0
        maxit = len(list(permutations(self.bestnode.actmove)))
```

```
        for i in permutations(self.bestnode.actmove):
            if it % 1000 == 0:
                print(it / maxit)
            it += 1
            ord = self.bestnode.premove + list(i)
            if self.isLegalMove(ord):
                self.savebest(self.fitfunc(ord), ord, "Bruteforce")


    def isLegalMove(self, ord):
        for i in range(len(ord)):
            f = self.defLM(self.con, [ord[i]], i, ord[:i])
            if not f:
                break
        return f


class node(object):
    def __init__(self, master, actmove, premove, movecount, predecessor):
        self.master = master
        self.actmove = actmove
        self.premove = premove
        self.movecount = movecount
        self.predecessor = predecessor
        self.legMove = master.defLM(self.master.con, self.actmove, self.movecount,
                                    self.premove)
        self.move = self.legMove[:]
        self.successor = []
        self.nbestval = pow(10, 10)
        self.wi = 0
        self.si = 0
        self.ssi = 0
        self.spsi = 0
        self.spssi = 0
        self.UCB = 0

    def UCBc(self, mode):
        if mode == 1:
            if self.ssi == 0:
                return (self.master.C) * math.sqrt((math.log(self.predecessor.spsi) / self.si))
            else:
                return -(self.wi / self.ssi) / self.master.bestval + (self.master.C) * math.sqrt(
                    (math.log(self.predecessor.spsi) / self.si))
        elif mode == 2:
            if self.ssi == 0:
                return (self.master.C) * math.sqrt((math.log(self.master.bestnode.spsi) / self.si))
            else:
                return -(self.wi / self.ssi) / self.master.bestval + (self.master.C) * math.sqrt(
                    (math.log(self.master.bestnode.spsi) / self.si))
        elif mode == 3:
            if self.ssi == 0:
                return (self.master.C) * math.sqrt((math.log(self.master.root.spsi) / self.si))
            else:
                return -(self.wi / self.ssi) / self.master.bestval + (self.master.C) * math.sqrt(
                    (math.log(self.master.root.spsi) / self.si))
        elif mode == 4:
            return -self.nbestval / self.master.bestval + (self.master.C) * math.sqrt(
                (math.log(self.predecessor.spsi) / self.si))
        elif mode == 5:
            return -self.nbestval / self.master.bestval + (self.master.C) * math.sqrt(
                (math.log(self.master.bestnode.spsi) / self.si))
        elif mode == 6:
            return -self.nbestval / self.master.bestval + (self.master.C) * math.sqrt(
                (math.log(self.master.root.spsi) / self.si))


    def pr(self):
        print("*" * 50)
        print("actmove: %s" % self.actmove)
        print("premove: %s" % self.premove)
        print("predecessor: %s" % self.predecessor)
        print("legMove: %s" % self.legMove)
        print("move: %s" % self.move)
        print("successor: %s" % self.successor)
        print("movecount %s" % self.movecount)
        print("wi: %s" % self.wi)
        print("si: %s" % self.si)
        print("UCB: %s" % self.UCB)

def recon(constrains):
    con1 = []
```

```
    con2 = []
    con3 = []
    con4 = []
    con5 = []
    for i in constrains:
        if re.match("[0-9]+=[0-9]+", i):
            con1.append(re.split("=", i))
        elif re.match("[0-9]+>>[0-9]+", i):
            con2.append(re.split(">>", i))
        elif re.match("[0-9]+>[0-9]+", i):
            con3.append(re.split(">", i))
        elif re.match("[0-9]+<>[0-9]+", i):
            con4.append(re.split("<>", i))
        elif re.match("[0-9]+_[0-9]+", i):
            con5.append(re.split("_", i))
        else:
            print("Złe ograniczenie")
    con1 = [list(map(int, i)) for i in con1]
    con2 = [list(map(int, i)) for i in con2]
    con3 = [list(map(int, i)) for i in con3]
    con4 = [list(map(int, i)) for i in con4]
    con5 = [list(map(int, i)) for i in con5]
    return [con1, con2, con3, con4, con5]


def defLegMove(con, actmove, movecount, premove):
    lm = []

    for i in actmove:
        legal = True
        for j in con[0]:
            if (j[1] != movecount) == (j[0] == i):
                legal = False

        if premove:
            for j in con[1]:
                if (j[0] != premove[-1]) == (j[1] == i):
                    legal = False
        else:
            for j in con[1]:
                if j[1] == i:
                    legal = False

        if premove:
            for j in con[2]:
                if (j[0] not in premove) and (j[1] == i):
                    legal = False
        else:
            for j in con[2]:
                if j[1] == i:
                    legal = False

        if premove:
            for j in con[3]:
                if (j[0] == premove[-1] and j[1] != i and j[1] not in premove) or (
                        j[1] == premove[-1] and j[0] != i and j[0] not in premove):
                    legal = False


        if premove:
            for j in con[4]:
                if not set(j).issubset(premove):
                    if (premove[-1] in j) and i not in j:
                        legal = False
        if legal:
            lm.append(i)
    return (lm)


def defLegMoveSJ1(con, actmove, movecount, premove):
    lm = []

    O110 = range(1, 21)
    O120 = range(21, 31)
    O150 = range(31, 41)

    O = [O110, O120, O150]

    for i in O:
        b = sorted(list(set(set(i) - set(premove))))
        if b:
```

```
            lm.append(b[0])

    return (lm)


def defLegMoveSJ2(con, actmove, movecount, premove):
    lm = []

    O110 = range(1, 21)
    O120 = range(21, 31)
    O150 = range(31, 41)

    B = [O120, O150]

    O = [O110, O120, O150]

    for i in B:
        if len(list(set(set(i) - set(premove)))) % 2 == 1:
            b = sorted(list(set(set(i) - set(premove))))
            return [b[0]]

    for i in O:
        b = sorted(list(set(set(i) - set(premove))))
        if b:
            lm.append(b[0])
    return (lm)


def defLegMoveSJ3(con, actmove, movecount, premove):
    lm = []

    O110 = range(1, 21)
    O120 = range(21, 31)
    O150 = range(31, 41)

    B = [O110, O120, O150]

    O = [O110, O120, O150]

    for i in B:
        if len(list(set(set(i) - set(premove)))) % 2 == 1:
            b = sorted(list(set(set(i) - set(premove))))
            return [b[0]]

    for i in O:
        b = sorted(list(set(set(i) - set(premove))))
        if b:
            lm.append(b[0])
    return (lm)
```

```
from PS import PS

'''Stałe'''
L1 = 1e4
L2 = 1e6
L3 = 1e8
L4 = 1e10
L5 = 1e12
L6 = 1e14
s=2
'''Typ efektu C1'''
name = "C1"
tgr = [[7, 8, 6, 7],
       [9, 4, 7, 9],
       [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
kindir = 1
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
        [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
        [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
        [0, 0, 0],
        [0, 0, 0]]
Cwbl = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
        [L1, L1, L1],
        [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca=[[0, L2, 0, L2, "NWR", 1, 1, "NWR", 0, 2],
    [0, L2, 0, L2, "NWR", 1, 1, "NWR", 2, 0],]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])

'''Typ efektu C2'''
name = "C2"
tgr = [[7, 8, 6, 7],
       [9, 4, 7, 9],
       [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
       [0, 0, 0, 0],
       [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
     [[0], [0], [0], [0]],
     [[0], [0], [0], [0]]]
kindir = 1
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
```

```
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1],
      [L1, L1, L1, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Ctbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L1, L1, L1],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca=[[0, L2, 0, L2, "NWZ", 1, 1, "NWZ", 0, 2],
   [0, L2, 0, L2, "NWZ", 1, 1, "NWZ", 2, 0],]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])
```

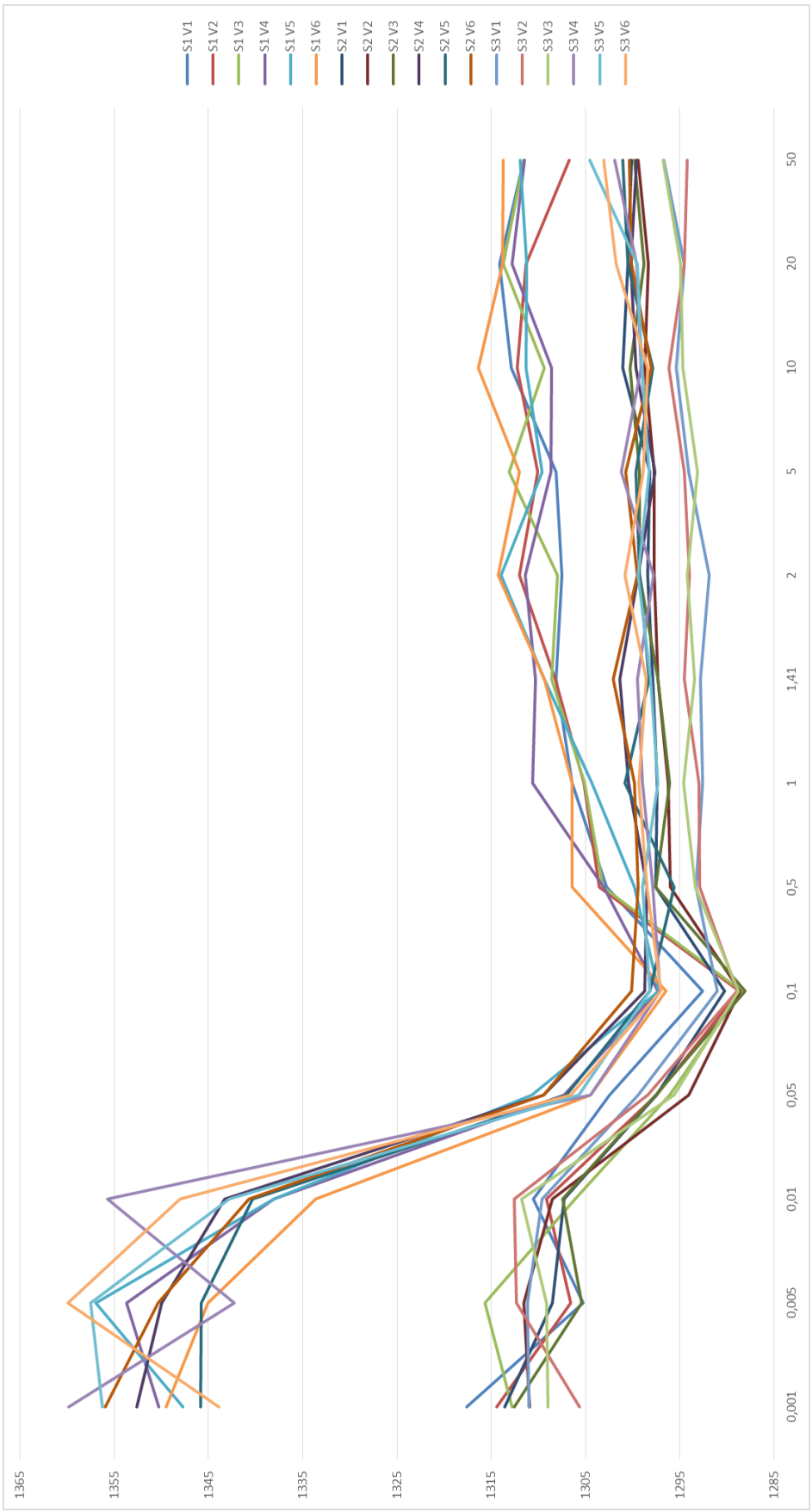
'''Typ efektu C3'''

```
name = "C3"
tgr = [[7, 8, 6, 7],
     [9, 4, 7, 9],
     [10, 7, 7, 4]]
kgr = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
t = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
a = [[[0], [0], [0], [0]],
    [[0], [0], [0], [0]],
    [[0], [0], [0], [0]]]
kindir = 1
kp = [0, 0, 0]
kn = [0, 0, 0]
kc = [0, 0, 0, 0]
Td = [0, 0, 0]
Tso = [0, 0, 0]
Tfo = [1000, 1000, 1000]
Tsb = [0, 0, 0, 0]
Tfb = [1000, 1000, 1000, 1000]
Ctol = [[0, 0, 0, 0],
      [0, 0, 0, 0]]
Cwol = [[L6, L6, L6, L6],
      [L6, L6, L6, L6]]
Ctou = [[L1,  0,  0, L1],
      [L1,  0,  0, L1]]
Cwou = [[L1, L2, L4, L1],
      [L1, L2, L4, L1]]
Ctbl = [[0, 0, 0],
      [0, 0, 0],
      [0, 0, 0]]
Cwbl = [[L6, L6, L6],
      [L6, L6, L6],
      [L6, L6, L6]]
Ctbu = [[L1, L1, L1],
      [ 0,  0,  0],
      [L1, L1, L1]]
Cwbu = [[L1, L1, L1],
      [L3, L3, L3],
      [L1, L1, L1]]
Co = [Ctol, Cwol, Ctou, Cwou]
Cb = [Ctbl, Cwbl, Ctbu, Cwbu]
Ca=[[0, L5, 0, L5, "NWZ", 0, 2, "NWZ", 1, 1],
   [0, L5, 0, L5, "NWZ", 1, 1, "NWZ", 2, 0]]
t = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl, Ctbu, Cwbu, Ca, pr=1, save=1)
t.solve([1, 2, 3])
```

Załącznik 7: Wartość średniej wartości funkcji celu w zależności od parametru C dla wszystkich kombinacji V_i oraz S_i

C	S1 V1	S1 V2	S1 V3	S1 V4	S1 V5	S1 V6	S2 V1	S2 V2	S2 V3	S2 V4	S2 V5	S2 V6	S3 V1	S3 V2	S3 V3	S3 V4	S3 V5	S3 V6
0,001	1317,65	1314,4	1312,8	1350,3	1347,75	1349,5	1313,55	1310,9	1312,55	1352,65	1345,85	1356	1311	1305,65	1309	1359,85	1356,3	1343,9
0,005	1305,25	1306,6	1315,7	1353,7	1357	1345,1	1308,55	1311,55	1305,4	1349,95	1345,8	1350,4	1311,2	1312,3	1309,15	1342,3	1357,5	1359,95
0,01	1310,55	1309,15	1306,15	1338,1	1338,1	1333,65	1307,3	1308,55	1307,4	1343,3	1340,35	1340,75	1309,65	1312,55	1311,8	1355,75	1342,8	1348,05
0,05	1302,5	1297,45	1296,05	1307,25	1310,7	1304,5	1297,5	1294,1	1297,55	1309,45	1307,05	1309,45	1299,4	1298,4	1295,6	1304,5	1305,7	1306,55
0,1	1292,6	1288,7	1288,5	1297,3	1297,25	1296,45	1290,25	1288,65	1288,1	1298,75	1298,2	1300,15	1291	1288,95	1288,75	1297,1	1298,15	1297,05
0,5	1302,75	1303,55	1303,15	1303,1	1299,8	1306,4	1297,55	1296	1297,55	1298,5	1295,6	1299,4	1293,3	1292,9	1293,35	1297,85	1298,95	1298,5
1	1306,45	1305,15	1305,1	1310,65	1304,3	1306,4	1297,4	1296,25	1296,1	1300,45	1300,85	1299,8	1292,55	1292,95	1294,6	1298,95	1297,35	1299,35
1,41	1308,1	1308,25	1308,6	1310,3	1309,4	1309,4	1297,9	1297,3	1297,3	1301,35	1298,3	1302,05	1292,8	1294,5	1293,45	1299,5	1298,1	1298,55
2	1307,5	1312	1308	1311,4	1314	1314,3	1298,45	1297,7	1299,25	1299,45	1299,3	1299,5	1291,85	1293,95	1294,2	1297,8	1299,45	1300,85
5	1308,1	1310,05	1313,15	1308,65	1309,65	1312	1298,2	1297,75	1299,25	1297,65	1299,65	1300,75	1294,05	1294,5	1293,1	1301,2	1298,25	1298,85
10	1312,85	1312,25	1309,35	1308,6	1311,35	1316,4	1301,05	1298,7	1300,3	1299,65	1297,9	1298	1295,35	1296,2	1294,65	1299,15	1299,05	1298,45
20	1314,1	1311,3	1313,75	1312,8	1311,25	1313,8	1300,5	1298,35	1298,8	1300,2	1300,55	1300,1	1294,5	1294,55	1294,9	1299,6	1299,5	1301,8
50	1311,55	1306,75	1311,55	1311,5	1311,95	1313,75	1300,15	1299,45	1299,8	1299,6	1301,05	1300,4	1296,7	1294,25	1296,75	1301,9	1304,55	1303,1

Załącznik 8: Wartość średniej wartości funkcji celu w zależności od parametru C dla wszystkich kombinacji V_i oraz S_i



Załącznik 9: Przebieg 20 różnych iteracji algorytmu MCTS dla (S_2, V_3) oraz wartości parametru $C = 0{,}1$.

Lp.	Czas działania programu	Funkcja celu	Węzeł	Iteracja	Uszeregowanie
1	0,000498772	1467	0	1	[11, 13, 14, 3, 12, 15, 8, 4, 2, 9, 18, 19, 7, 6, 5, 1, 10, 16, 17, 20]
	0,002494574	1435	0	6	[9, 14, 1, 10, 15, 11, 5, 6, 12, 8, 18, 13, 17, 20, 16, 3, 7, 2, 4, 19]
	0,002993584	1428	0	8	[2, 12, 5, 1, 6, 15, 9, 4, 19, 17, 8, 14, 13, 7, 11, 10, 16, 3, 20, 18]
	0,005489111	1376	0	15	[17, 8, 19, 14, 15, 3, 16, 7, 4, 18, 5, 12, 9, 6, 13, 11, 20, 2, 1, 10]
	0,04391551	1369	0	114	[6, 3, 4, 11, 13, 14, 2, 7, 17, 1, 10, 12, 9, 15, 8, 19, 18, 5, 16, 20]
	0,138738871	1368	0	350	[9, 5, 4, 15, 16, 3, 12, 11, 8, 17, 13, 19, 14, 18, 20, 6, 1, 2, 10, 7]
	0,167685509	1335	0	420	[6, 12, 15, 11, 8, 10, 7, 9, 17, 2, 14, 1, 16, 5, 19, 13, 4, 3, 18, 20]
	0,340363979	1334	0	812	[15, 14, 8, 9, 17, 18, 2, 1, 13, 19, 6, 12, 16, 3, 4, 7, 11, 10, 5, 20]
	0,885849714	1324	0	2046	[3, 9, 13, 1, 8, 17, 6, 19, 16, 15, 7, 11, 5, 14, 10, 4, 12, 2, 18, 20]
	3,893238068	1311	0	7455	[15, 9, 13, 14, 6, 19, 12, 4, 3, 5, 17, 16, 11, 1, 8, 18, 2, 10, 7, 20]
	9,203380346	1298	0	17325	[15, 6, 4, 3, 11, 13, 5, 9, 14, 17, 7, 18, 1, 19, 12, 16, 8, 10, 2, 20]
	14,46160078	1297	0	26669	[15, 6, 9, 13, 14, 19, 1, 8, 17, 16, 5, 4, 10, 7, 11, 2, 18, 12, 3, 20]
	35,40814161	1294	1	60540	[15, 6, 9, 13, 11, 14, 5, 17, 4, 2, 7, 8, 3, 19, 18, 16, 1, 10, 20, 12]
	64,71162343	1292	11	104281	[15, 6, 9, 13, 11, 14, 5, 17, 4, 2, 7, 19, 1, 18, 8, 16, 10, 20, 12, 3]
	65,77216721	1287	Bruteforce	106467	[15, 6, 9, 13, 11, 14, 5, 17, 4, 2, 7, 19, 1, 8, 18, 16, 10, 20, 12, 3]
2	0,000498772	1601	0	1	[3, 12, 13, 2, 10, 4, 15, 20, 18, 6, 1, 17, 14, 19, 9, 8, 5, 16, 7, 11]
	0,000997305	1515	0	2	[19, 18, 9, 13, 11, 10, 16, 20, 1, 8, 6, 2, 4, 15, 12, 3, 5, 14, 7, 17]
	0,001498222	1484	0	3	[10, 7, 18, 3, 12, 15, 6, 19, 14, 8, 5, 9, 1, 20, 16, 11, 13, 17, 2, 4]
	0,003493071	1480	0	9	[5, 13, 2, 11, 3, 20, 8, 10, 14, 16, 6, 7, 4, 15, 17, 12, 1, 19, 18, 9]
	0,00597167	1356	0	15	[15, 7, 6, 2, 17, 9, 18, 3, 8, 19, 12, 1, 14, 11, 5, 16, 13, 4, 20, 10]
	0,251015663	1352	0	614	[7, 9, 6, 15, 17, 1, 13, 19, 2, 16, 14, 4, 18, 3, 12, 5, 20, 8, 11, 10]
	0,862875938	1342	0	2006	[15, 14, 11, 7, 3, 17, 9, 4, 19, 12, 10, 1, 5, 8, 2, 13, 16, 6, 18, 20]
	1,766696215	1339	0	3719	[15, 1, 9, 17, 3, 4, 7, 13, 11, 14, 18, 19, 8, 12, 5, 6, 2, 16, 20, 10]
	3,279382229	1332	0	6456	[15, 6, 5, 13, 1, 19, 2, 9, 4, 11, 17, 14, 18, 7, 3, 16, 20, 8, 10, 12]
	3,564850092	1316	0	6917	[15, 6, 4, 19, 14, 5, 2, 17, 1, 16, 10, 7, 11, 3, 13, 8, 9, 18, 12, 20]
	5,857587576	1299	0	10333	[15, 6, 17, 4, 3, 11, 9, 13, 18, 7, 16, 1, 2, 19, 8, 14, 5, 10, 12, 20]
	12,03509665	1297	0	21546	[15, 6, 9, 13, 17, 11, 8, 3, 16, 5, 14, 7, 19, 4, 2, 1, 18, 12, 10, 20]
	53,20903659	1294	3	87530	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 8, 19, 1, 3, 2, 16, 10, 20, 12]
	54,83551097	1293	3	90134	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 16, 1, 19, 10, 2, 8, 20, 12, 3]
	55,119982	1287	3	90589	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 16, 8, 19, 1, 2, 10, 20, 12, 3]
3	0,000498533	1502	0	1	[7, 4, 1, 17, 20, 11, 5, 16, 6, 13, 3, 8, 9, 14, 19, 18, 15, 10, 2, 12]
	0,001496792	1486	0	4	[14, 19, 15, 4, 11, 8, 9, 12, 10, 16, 5, 17, 13, 6, 7, 2, 20, 3, 18, 1]
	0,002993822	1453	0	7	[13, 18, 17, 20, 9, 15, 12, 3, 7, 1, 5, 14, 8, 16, 10, 4, 19, 2, 11, 6]
	0,004990101	1419	0	14	[11, 1, 17, 15, 2, 14, 16, 8, 13, 9, 20, 19, 12, 4, 6, 10, 3, 5, 18, 7]
	0,020962238	1405	0	54	[14, 6, 13, 15, 5, 7, 4, 9, 16, 11, 12, 3, 1, 10, 2, 17, 20, 8, 19, 18]
	0,022457361	1399	0	58	[7, 1, 8, 19, 11, 17, 16, 13, 4, 5, 3, 10, 14, 12, 9, 6, 18, 2, 20, 15]
	0,052901506	1397	0	135	[8, 3, 12, 15, 2, 17, 7, 11, 18, 19, 13, 16, 4, 6, 5, 14, 1, 9, 20, 10]
	0,057893515	1382	0	148	[13, 19, 15, 6, 8, 5, 3, 7, 1, 12, 14, 16, 20, 11, 10, 2, 17, 4, 9, 18]
	0,123768091	1377	0	315	[8, 3, 14, 17, 10, 4, 6, 19, 11, 13, 12, 15, 7, 5, 9, 2, 20, 16, 1, 18]
	0,151200294	1370	0	382	[3, 15, 17, 9, 13, 8, 4, 7, 6, 11, 2, 5, 16, 12, 1, 14, 20, 19, 18, 10]
	0,362323999	1351	0	877	[13, 17, 2, 6, 4, 15, 11, 12, 19, 9, 1, 8, 16, 5, 3, 14, 18, 7, 10, 20]
	0,363321543	1344	0	880	[13, 6, 4, 1, 11, 14, 2, 16, 15, 9, 3, 5, 7, 8, 19, 12, 18, 17, 20, 10]
	1,50368309	1341	0	3298	[15, 14, 9, 6, 17, 16, 11, 8, 7, 5, 18, 12, 13, 3, 1, 4, 19, 20, 2, 10]
	2,193897724	1339	0	4602	[13, 4, 9, 3, 1, 17, 8, 2, 5, 7, 12, 6, 16, 19, 15, 11, 14, 18, 20, 10]
	3,722076416	1334	0	7412	[15, 6, 9, 19, 14, 5, 11, 7, 4, 8, 2, 17, 18, 20, 16, 3, 13, 1, 10, 12]
	4,092366219	1326	0	8065	[15, 6, 9, 1, 8, 13, 5, 16, 4, 3, 19, 12, 7, 17, 2, 10, 14, 11, 18, 20]
	4,903857946	1324	0	9484	[15, 6, 13, 17, 4, 14, 11, 19, 18, 8, 9, 16, 1, 3, 5, 2, 10, 7, 12, 20]
	7,142213583	1307	0	13669	[15, 6, 9, 7, 14, 11, 19, 8, 1, 5, 4, 10, 18, 12, 2, 17, 13, 16, 3, 20]
	9,4918437	1297	0	17976	[15, 6, 9, 4, 1, 8, 11, 5, 17, 7, 2, 18, 19, 13, 12, 3, 14, 16, 10, 20]
	50,03493786	1296	2	82776	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 16, 3, 8, 19, 7, 1, 2, 10, 20, 12]
	50,65079212	1294	2	83718	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 16, 1, 2, 3, 8, 19, 10, 20, 12]
	51,99627042	1293	2	85746	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 1, 2, 19, 8, 16, 10, 20, 12, 3]
	52,77382469	1287	3	86899	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 1, 19, 8, 16, 2, 10, 20, 12, 3]
4	0,000499487	1414	0	1	[13, 11, 4, 8, 15, 9, 1, 5, 18, 3, 6, 19, 7, 20, 17, 16, 10, 14, 12, 2]
	0,018465519	1392	0	50	[6, 11, 17, 16, 3, 4, 15, 8, 14, 9, 18, 2, 1, 12, 20, 10, 19, 5, 7, 13]
	0,020462275	1388	0	55	[15, 6, 5, 3, 16, 17, 7, 11, 18, 10, 9, 8, 4, 12, 19, 14, 1, 2, 20, 13]
	0,033438206	1378	0	88	[13, 6, 15, 9, 1, 5, 11, 12, 10, 7, 3, 17, 14, 4, 20, 8, 16, 2, 19, 18]
	0,039925814	1367	0	104	[8, 11, 6, 18, 9, 4, 1, 15, 16, 3, 19, 13, 12, 7, 17, 20, 5, 2, 14, 10]
	0,203122139	1366	0	445	[19, 6, 15, 3, 1, 13, 14, 2, 12, 9, 5, 16, 17, 18, 4, 11, 7, 8, 20, 10]
	0,374303818	1350	0	843	[13, 4, 2, 3, 9, 17, 8, 14, 5, 12, 6, 1, 16, 15, 7, 18, 19, 11, 20, 10]
	0,605873108	1347	0	1351	[6, 14, 17, 3, 15, 12, 13, 1, 8, 9, 11, 2, 7, 10, 4, 5, 19, 18, 16, 20]
	0,979159594	1339	0	2060	[15, 6, 13, 16, 14, 5, 11, 17, 7, 8, 3, 4, 19, 12, 18, 1, 9, 20, 2, 10]
	1,949853659	1325	0	3889	[15, 4, 9, 19, 3, 7, 11, 6, 5, 14, 2, 1, 13, 8, 16, 10, 17, 12, 18, 20]
	2,307190895	1324	0	4556	[15, 6, 13, 8, 2, 17, 9, 12, 5, 4, 10, 14, 16, 19, 7, 1, 3, 11, 18, 20]
	3,51693964	1301	0	6625	[15, 6, 4, 11, 17, 14, 5, 19, 7, 10, 3, 8, 16, 1, 18, 12, 9, 2, 13, 20]
	16,32014656	1297	0	27463	[15, 6, 9, 14, 8, 16, 19, 3, 1, 4, 17, 2, 7, 5, 18, 13, 12, 11, 10, 20]
	40,14980793	1295	1	65730	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 1, 3, 16, 18, 19, 8, 2, 10, 20, 12]
	47,94183469	1294	2	77878	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 8, 16, 18, 1, 19, 2, 10, 3, 20, 12]
	51,19727802	1287	2	82953	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 8, 16, 19, 1, 2, 10, 20, 12, 3]

5	0,000498772	1514	0	1	[12, 14, 15, 7, 17, 4, 18, 19, 16, 11, 9, 1, 2, 13, 20, 3, 5, 8, 6, 10]
	0,001496553	1430	0	5	[6, 10, 1, 16, 17, 3, 13, 5, 15, 8, 9, 12, 19, 2, 14, 18, 4, 7, 20, 11]
	0,00199604	1409	0	6	[18, 14, 17, 6, 9, 16, 15, 11, 12, 7, 20, 5, 1, 3, 19, 8, 4, 10, 2, 13]
	0,012476921	1406	0	35	[8, 17, 6, 3, 4, 14, 11, 7, 12, 9, 1, 16, 5, 15, 13, 18, 19, 20, 10, 2]
	0,025951147	1402	0	70	[5, 6, 8, 19, 4, 15, 14, 9, 13, 16, 18, 2, 7, 20, 17, 12, 3, 11, 10, 1]
	0,028945923	1386	0	77	[8, 5, 9, 1, 17, 14, 4, 11, 18, 6, 2, 20, 15, 19, 16, 7, 12, 3, 10, 13]
	0,035433292	1364	0	93	[8, 1, 15, 4, 19, 18, 11, 14, 5, 7, 20, 9, 17, 10, 3, 13, 12, 16, 2, 6]
	0,160700798	1360	0	402	[11, 1, 4, 15, 8, 16, 5, 7, 6, 17, 9, 14, 3, 18, 12, 2, 10, 20, 19, 13]
	0,295949459	1359	0	720	[7, 11, 15, 1, 14, 16, 5, 4, 17, 18, 6, 13, 9, 19, 3, 12, 8, 2, 20, 10]
	0,617853165	1346	0	1461	[17, 15, 19, 13, 9, 14, 3, 5, 1, 16, 7, 8, 4, 11, 20, 6, 18, 2, 12, 10]
	0,675245762	1344	0	1594	[15, 13, 17, 6, 14, 5, 8, 19, 16, 7, 18, 9, 1, 12, 3, 11, 2, 4, 10, 20]
	0,987663269	1342	0	2310	[15, 9, 14, 2, 6, 17, 4, 8, 19, 11, 5, 7, 16, 13, 12, 18, 1, 3, 20, 10]
	1,777673721	1325	0	3729	[15, 14, 9, 8, 13, 4, 16, 6, 17, 19, 12, 7, 11, 2, 5, 1, 3, 10, 18, 20]
	2,414988279	1297	0	4864	[15, 3, 4, 1, 9, 6, 17, 2, 7, 14, 11, 16, 13, 18, 8, 5, 19, 12, 10, 20]
6	0,000499487	1537	0	1	[2, 9, 4, 15, 18, 20, 6, 8, 10, 11, 17, 14, 13, 16, 3, 1, 12, 19, 7, 5]
	0,000998259	1506	0	2	[12, 9, 5, 15, 16, 10, 19, 3, 4, 7, 8, 1, 11, 2, 13, 20, 18, 6, 14, 17]
	0,001496553	1498	0	3	[19, 5, 16, 13, 17, 4, 18, 8, 11, 1, 7, 14, 6, 12, 3, 20, 10, 9, 2, 15]
	0,001496553	1449	0	4	[1, 16, 9, 10, 4, 2, 8, 17, 19, 18, 3, 13, 11, 5, 15, 7, 6, 14, 12, 20]
	0,002495289	1422	0	7	[11, 9, 19, 3, 20, 6, 5, 15, 16, 4, 17, 12, 13, 2, 1, 18, 10, 7, 8, 14]
	0,010480404	1394	0	29	[4, 14, 19, 1, 16, 8, 17, 5, 9, 15, 6, 13, 20, 7, 2, 12, 10, 3, 11, 18]
	0,016488314	1389	0	45	[17, 6, 9, 14, 8, 5, 1, 2, 18, 16, 20, 7, 11, 15, 19, 10, 12, 3, 4, 13]
	0,046415329	1370	0	120	[14, 4, 17, 16, 13, 20, 11, 3, 8, 6, 15, 2, 9, 12, 1, 19, 5, 18, 10, 7]
	0,155230284	1362	0	392	[5, 15, 3, 14, 17, 12, 2, 1, 19, 4, 11, 13, 18, 9, 8, 6, 7, 16, 20, 10]
	0,560976028	1348	0	1328	[15, 8, 3, 9, 6, 1, 5, 13, 2, 4, 7, 19, 17, 11, 16, 18, 14, 20, 10, 12]
	0,640826941	1340	0	1511	[13, 11, 6, 15, 12, 5, 8, 4, 16, 19, 3, 14, 9, 17, 1, 7, 2, 18, 10, 20]
	0,708201647	1335	0	1667	[3, 15, 9, 1, 8, 11, 6, 13, 14, 17, 10, 19, 4, 7, 5, 16, 18, 20, 12, 2]
	2,560238123	1329	0	5127	[15, 14, 3, 6, 13, 11, 5, 17, 19, 16, 18, 12, 1, 8, 9, 10, 7, 4, 2, 20]
	3,516459465	1327	0	6775	[15, 13, 3, 11, 1, 4, 6, 17, 14, 9, 16, 18, 5, 8, 7, 19, 10, 20, 12, 2]
	4,625895739	1311	0	8676	[15, 6, 9, 4, 11, 13, 1, 5, 7, 17, 8, 16, 19, 2, 18, 3, 12, 14, 10, 20]
	7,672748327	1307	0	14203	[15, 6, 14, 16, 13, 17, 11, 5, 19, 8, 3, 4, 18, 9, 1, 2, 7, 10, 20, 12]
	7,768070936	1297	0	14383	[15, 6, 3, 1, 17, 14, 19, 8, 2, 11, 13, 7, 9, 16, 4, 5, 10, 18, 12, 20]
7	0,000498772	1534	0	1	[20, 2, 15, 14, 11, 12, 16, 3, 13, 4, 10, 8, 7, 6, 17, 5, 9, 18, 1, 19]
	0,000998259	1523	0	2	[9, 16, 14, 7, 11, 12, 2, 15, 20, 1, 8, 5, 13, 17, 18, 3, 10, 4, 19, 6]
	0,001995802	1501	0	5	[15, 9, 6, 19, 1, 11, 14, 3, 5, 8, 10, 2, 12, 16, 18, 7, 20, 13, 17, 4]
	0,002494574	1495	0	7	[7, 18, 20, 12, 11, 15, 1, 13, 6, 4, 17, 19, 16, 8, 14, 5, 10, 9, 2, 3]
	0,002993822	1453	0	8	[6, 8, 14, 11, 15, 4, 7, 10, 19, 17, 20, 18, 3, 2, 1, 16, 9, 5, 13, 12]
	0,003493309	1372	0	9	[13, 5, 11, 15, 9, 6, 4, 16, 19, 7, 8, 14, 1, 2, 10, 17, 18, 3, 12, 20]
	0,036930799	1325	0	96	[15, 17, 13, 9, 18, 11, 7, 1, 6, 16, 5, 2, 3, 14, 4, 8, 19, 12, 10, 20]
	0,068372488	1311	0	177	[13, 4, 15, 11, 8, 5, 9, 16, 17, 6, 19, 7, 14, 2, 1, 18, 12, 10, 3, 20]
	4,349390984	1305	0	8225	[15, 6, 5, 8, 17, 9, 19, 11, 13, 4, 16, 12, 3, 1, 2, 14, 18, 10, 7, 20]
	7,641287804	1304	0	14292	[15, 6, 9, 5, 14, 1, 19, 7, 16, 11, 3, 13, 18, 2, 12, 17, 8, 4, 10, 20]
	7,789012671	1299	0	14566	[15, 6, 9, 3, 8, 14, 4, 17, 1, 7, 11, 5, 18, 12, 10, 2, 16, 19, 13, 20]
	8,923902035	1297	0	16610	[15, 6, 9, 14, 17, 5, 1, 11, 7, 13, 3, 19, 8, 4, 18, 12, 16, 2, 10, 20]
	14,60034442	1294	0	26613	[15, 6, 9, 13, 8, 4, 17, 5, 14, 19, 11, 1, 2, 3, 7, 18, 16, 10, 20, 12]
	54,83800673	1293	3	88760	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 19, 8, 16, 1, 2, 10, 20, 12, 3]
	57,47659874	1287	4	92949	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 1, 16, 19, 8, 2, 10, 20, 12, 3]
8	0,000498533	1560	0	1	[2, 1, 9, 19, 5, 6, 4, 14, 10, 12, 7, 18, 11, 8, 13, 15, 3, 17, 20, 16]
	0,000997066	1516	0	2	[20, 9, 7, 15, 14, 10, 13, 19, 8, 5, 11, 1, 16, 17, 3, 6, 12, 4, 18, 2]
	0,001496553	1504	0	4	[13, 20, 17, 4, 10, 9, 8, 6, 16, 5, 15, 2, 7, 14, 18, 19, 3, 1, 11, 12]
	0,003492594	1441	0	10	[6, 9, 5, 14, 13, 18, 8, 16, 7, 1, 4, 19, 17, 10, 12, 15, 3, 20, 2, 11]
	0,004989624	1423	0	14	[14, 19, 17, 13, 7, 1, 9, 3, 10, 11, 18, 2, 15, 8, 20, 16, 4, 6, 12, 5]
	0,005489111	1390	0	15	[15, 14, 4, 5, 6, 18, 9, 2, 12, 16, 19, 17, 1, 8, 3, 11, 13, 7, 20, 10]
	0,071847439	1364	0	186	[17, 11, 13, 5, 6, 15, 4, 3, 16, 14, 7, 9, 18, 8, 20, 2, 1, 12, 19, 10]
	0,204119682	1330	0	509	[15, 3, 7, 11, 6, 13, 4, 19, 17, 16, 5, 14, 8, 9, 1, 18, 2, 20, 10, 12]
	3,938654661	1310	0	7679	[15, 6, 1, 5, 9, 17, 8, 3, 13, 7, 16, 4, 11, 19, 2, 18, 10, 12, 14, 20]
	8,290579796	1297	0	15464	[15, 6, 9, 14, 16, 19, 8, 17, 5, 4, 13, 1, 3, 11, 2, 7, 10, 18, 12, 20]
	42,14561272	1295	1	70059	[15, 6, 9, 13, 11, 14, 4, 5, 19, 17, 1, 18, 16, 7, 8, 2, 10, 3, 20, 12]
	50,08384871	1287	2	82249	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 8, 19, 1, 16, 2, 10, 20, 12, 3]
9	0,00049901	1499	0	1	[5, 1, 17, 10, 13, 15, 14, 12, 3, 4, 20, 19, 7, 16, 2, 6, 9, 11, 8, 18]
	0,000997782	1451	0	2	[2, 9, 14, 1, 19, 15, 17, 12, 6, 10, 8, 13, 16, 18, 11, 5, 20, 7, 3, 4]
	0,003493071	1429	0	10	[8, 5, 6, 10, 16, 17, 11, 15, 2, 7, 14, 12, 9, 3, 18, 4, 19, 20, 13, 1]
	0,030943394	1362	0	80	[17, 1, 19, 15, 6, 18, 8, 9, 13, 4, 12, 2, 11, 3, 14, 16, 10, 7, 5, 20]
	0,33289957	1341	0	802	[13, 9, 4, 2, 7, 11, 5, 15, 8, 16, 6, 17, 1, 3, 20, 14, 18, 12, 19, 10]
	1,049546719	1339	0	2398	[13, 9, 4, 6, 5, 19, 15, 12, 7, 17, 11, 14, 1, 2, 8, 18, 3, 16, 20, 10]
	1,70732379	1338	0	3664	[15, 3, 4, 13, 9, 6, 19, 11, 7, 17, 14, 5, 1, 16, 18, 8, 2, 12, 10, 20]
	2,072644234	1330	0	4334	[15, 17, 8, 14, 2, 9, 13, 3, 19, 7, 4, 11, 1, 5, 16, 12, 6, 18, 10, 20]
	2,079630852	1321	0	4349	[15, 14, 4, 11, 13, 17, 5, 1, 16, 18, 9, 6, 7, 2, 12, 10, 19, 3, 8, 20]
	4,133311987	1310	0	8044	[15, 9, 14, 8, 17, 6, 7, 3, 19, 16, 4, 13, 2, 1, 5, 18, 12, 11, 10, 20]
	5,232268095	1303	0	9946	[15, 6, 9, 19, 3, 17, 14, 18, 12, 5, 7, 11, 1, 8, 2, 13, 16, 4, 10, 20]
	8,051545858	1297	0	12711	[15, 6, 9, 17, 13, 1, 19, 4, 14, 5, 18, 16, 7, 3, 12, 8, 11, 10, 2, 20]
	48,09855962	1295	2	78020	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 1, 2, 18, 8, 16, 3, 19, 10, 20, 12]
	49,81187344	1294	2	80709	[15, 6, 9, 13, 11, 14, 4, 5, 17, 19, 8, 7, 18, 1, 3, 16, 2, 10, 20, 12]
	52,71148086	1293	3	85144	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 19, 16, 8, 1, 2, 10, 20, 12, 3]

	55,69094038	1287	3	89808	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 1, 16, 2, 8, 19, 10, 20, 12, 3]
10	0	1534	0	1	[6, 19, 20, 9, 5, 7, 1, 16, 12, 18, 8, 4, 11, 13, 14, 15, 10, 17, 3, 2]
	0,000499487	1413	0	2	[4, 12, 13, 15, 1, 19, 17, 16, 20, 8, 2, 6, 14, 11, 10, 9, 7, 5, 3, 18]
	0,006489277	1375	0	20	[15, 2, 6, 3, 17, 1, 18, 19, 9, 12, 5, 4, 11, 13, 7, 16, 8, 10, 20, 14]
	0,300441742	1372	0	719	[6, 20, 2, 8, 17, 9, 13, 3, 7, 12, 15, 19, 16, 14, 4, 1, 11, 5, 18, 10]
	0,42870307	1360	0	1011	[19, 3, 8, 15, 5, 9, 6, 11, 2, 1, 7, 17, 12, 14, 10, 16, 13, 4, 18, 20]
	0,549478531	1353	0	1290	[13, 6, 4, 15, 16, 1, 9, 20, 19, 2, 14, 17, 3, 5, 8, 18, 7, 12, 11, 10]
	1,004632473	1350	0	2311	[6, 9, 15, 14, 8, 20, 5, 13, 12, 17, 11, 16, 1, 18, 19, 3, 4, 2, 10, 7]
	1,172302246	1333	0	2622	[15, 7, 1, 16, 6, 14, 17, 2, 12, 4, 9, 3, 8, 11, 19, 13, 5, 18, 10, 20]
	1,719285011	1324	0	3671	[15, 9, 1, 3, 4, 12, 11, 13, 6, 17, 8, 2, 14, 5, 16, 19, 7, 10, 18, 20]
	1,954847574	1313	0	4105	[15, 9, 1, 5, 14, 8, 4, 19, 6, 7, 11, 16, 3, 18, 12, 10, 17, 2, 13, 20]
	2,433954716	1304	0	4883	[15, 9, 14, 17, 19, 6, 7, 4, 5, 11, 13, 16, 3, 1, 18, 2, 8, 12, 10, 20]
	11,33940983	1297	0	21085	[15, 6, 9, 13, 16, 17, 7, 11, 2, 1, 14, 5, 4, 3, 18, 19, 8, 12, 10, 20]
	40,92787862	1294	1	67133	[15, 6, 9, 13, 11, 14, 4, 5, 19, 7, 8, 17, 3, 1, 2, 18, 16, 10, 20, 12]
	43,7586143	1287	1	71557	[15, 6, 9, 13, 11, 14, 4, 19, 17, 5, 8, 7, 1, 2, 18, 16, 10, 20, 12, 3]
11	0,000499487	1531	0	1	[17, 1, 2, 14, 5, 7, 19, 3, 20, 18, 10, 4, 13, 9, 15, 12, 16, 6, 8, 11]
	0,00099802	1478	0	2	[14, 15, 3, 12, 1, 20, 4, 11, 19, 7, 10, 6, 13, 8, 2, 17, 9, 18, 16, 5]
	0,001996279	1443	0	5	[15, 16, 9, 17, 1, 10, 19, 11, 13, 5, 8, 20, 14, 18, 4, 2, 6, 3, 12, 7]
	0,002495527	1427	0	7	[3, 18, 12, 15, 16, 19, 14, 8, 2, 1, 9, 20, 6, 4, 17, 5, 13, 7, 11, 10]
	0,003493309	1413	0	10	[1, 9, 18, 3, 11, 15, 13, 4, 10, 20, 7, 14, 16, 19, 12, 6, 2, 17, 5, 8]
	0,024473429	1377	0	65	[15, 6, 9, 14, 10, 17, 3, 5, 4, 11, 16, 8, 1, 20, 13, 7, 19, 12, 2, 18]
	0,055416346	1371	0	143	[9, 15, 1, 11, 7, 4, 18, 6, 17, 2, 8, 13, 12, 19, 5, 16, 10, 3, 14, 20]
	0,120294571	1363	0	257	[6, 5, 3, 17, 11, 13, 14, 15, 19, 1, 9, 4, 8, 20, 16, 12, 18, 2, 10, 7]
	0,173695564	1338	0	362	[15, 11, 4, 1, 16, 6, 10, 3, 9, 13, 8, 2, 14, 12, 7, 5, 19, 17, 18, 20]
	0,392769575	1329	0	756	[6, 13, 9, 14, 17, 2, 11, 16, 8, 7, 18, 15, 12, 3, 19, 1, 5, 4, 10, 20]
	0,47613287	1319	0	890	[15, 6, 5, 4, 14, 16, 18, 17, 1, 19, 3, 2, 13, 12, 11, 8, 9, 7, 10, 20]
	3,453078985	1310	0	6517	[15, 6, 3, 19, 16, 5, 9, 8, 14, 17, 11, 13, 4, 18, 7, 2, 1, 12, 10, 20]
	6,443035364	1302	0	12227	[15, 6, 3, 14, 11, 5, 17, 8, 16, 1, 9, 4, 18, 2, 7, 19, 13, 12, 10, 20]
	8,534646273	1297	0	16089	[15, 6, 9, 14, 11, 2, 5, 13, 4, 1, 19, 8, 16, 18, 17, 7, 10, 12, 3, 20]
	42,68161654	1295	1	71396	[15, 6, 9, 13, 11, 14, 17, 4, 5, 7, 19, 1, 18, 16, 10, 2, 8, 3, 20, 12]
	45,4005785	1294	2	73307	[15, 6, 9, 13, 11, 14, 17, 4, 5, 19, 3, 2, 1, 18, 7, 16, 8, 10, 20, 12]
	45,89515972	1287	2	74108	[15, 6, 9, 13, 11, 14, 17, 4, 5, 7, 19, 8, 18, 2, 16, 1, 10, 20, 12, 3]
12	0,000498772	1610	0	1	[16, 5, 19, 7, 17, 10, 14, 8, 1, 15, 4, 3, 2, 13, 18, 6, 11, 20, 9, 12]
	0,000498772	1401	0	2	[9, 6, 15, 11, 16, 14, 17, 1, 3, 4, 7, 8, 18, 20, 5, 2, 19, 13, 12, 10]
	0,027430058	1397	0	74	[13, 2, 14, 7, 11, 1, 4, 3, 10, 17, 6, 9, 8, 12, 19, 15, 18, 16, 20, 5]
	0,044899225	1395	0	120	[9, 14, 6, 17, 11, 8, 20, 3, 5, 2, 15, 13, 16, 19, 4, 18, 1, 12, 7, 10]
	0,067374468	1386	0	178	[8, 13, 4, 19, 9, 3, 1, 10, 15, 11, 14, 17, 12, 16, 5, 18, 2, 7, 6, 20]
	0,129242659	1371	0	331	[15, 1, 6, 8, 3, 4, 9, 11, 17, 5, 19, 7, 10, 2, 14, 13, 18, 16, 12, 20]
	0,437186241	1351	0	1049	[15, 4, 17, 11, 1, 18, 6, 9, 5, 3, 12, 13, 16, 2, 7, 14, 19, 20, 8, 10]
	0,446170568	1310	0	1071	[15, 14, 12, 3, 17, 11, 19, 13, 4, 6, 8, 2, 18, 16, 1, 9, 5, 7, 10, 20]
	8,931388855	1305	0	16385	[15, 6, 9, 4, 11, 17, 3, 5, 14, 18, 12, 16, 13, 1, 2, 8, 19, 10, 7, 20]
	9,185397148	1303	0	16840	[15, 6, 9, 8, 16, 13, 7, 14, 11, 19, 3, 1, 18, 4, 17, 5, 2, 10, 12, 20]
	18,01000547	1297	0	31877	[15, 6, 9, 13, 11, 7, 8, 3, 17, 5, 14, 4, 2, 12, 18, 16, 19, 1, 10, 20]
	45,82077837	1294	2	75152	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 8, 19, 18, 1, 2, 16, 3, 10, 20, 12]
	52,76636076	1287	3	85798	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 8, 16, 19, 1, 2, 10, 20, 12, 3]
13	0,000498533	1507	0	1	[17, 7, 2, 1, 18, 6, 14, 19, 16, 13, 10, 5, 3, 9, 15, 12, 8, 4, 11, 20]
	0,00099802	1493	0	2	[16, 19, 1, 20, 11, 7, 10, 9, 14, 13, 17, 15, 6, 2, 4, 12, 3, 18, 8, 5]
	0,00099802	1363	0	3	[13, 1, 6, 12, 11, 15, 2, 16, 8, 19, 3, 9, 14, 18, 17, 7, 5, 20, 4, 10]
	0,529012918	1335	0	1254	[13, 9, 11, 6, 8, 2, 7, 1, 14, 15, 12, 19, 17, 18, 3, 16, 4, 10, 5, 20]
	1,081468344	1322	0	2472	[13, 11, 15, 6, 16, 8, 19, 2, 1, 17, 3, 4, 5, 9, 14, 10, 7, 18, 12, 20]
	3,266901493	1319	0	6574	[15, 6, 17, 14, 4, 12, 19, 7, 11, 9, 1, 3, 5, 18, 8, 2, 16, 13, 10, 20]
	8,439802408	1310	0	16110	[15, 6, 9, 4, 2, 8, 17, 14, 10, 13, 7, 11, 16, 5, 19, 1, 18, 20, 12, 3]
	8,617469072	1305	0	16442	[15, 6, 9, 14, 3, 1, 17, 2, 5, 11, 4, 18, 12, 19, 13, 8, 16, 10, 7, 20]
	12,6275115	1297	0	23594	[15, 6, 9, 13, 11, 14, 7, 8, 3, 1, 16, 17, 4, 2, 18, 19, 5, 12, 10, 20]
	36,83598566	1296	1	60637	[15, 6, 9, 13, 11, 14, 4, 17, 19, 8, 3, 1, 5, 7, 16, 18, 2, 10, 20, 12]
	50,3298893	1287	2	81937	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 1, 2, 18, 8, 16, 19, 10, 20, 12, 3]
14	0,000483274	1582	0	1	[5, 11, 3, 12, 20, 14, 4, 19, 16, 8, 10, 18, 9, 17, 2, 13, 7, 6, 15, 1]
	0,00099802	1483	0	2	[9, 16, 15, 18, 13, 12, 14, 2, 6, 20, 5, 11, 19, 17, 8, 7, 10, 1, 3, 4]
	0,001995564	1446	0	5	[3, 7, 9, 2, 6, 1, 11, 18, 13, 14, 16, 5, 8, 4, 19, 20, 15, 17, 12, 10]
	0,009481907	1403	0	27	[7, 4, 9, 17, 1, 3, 16, 10, 6, 19, 8, 2, 11, 13, 18, 12, 14, 15, 5, 20]
	0,009981155	1392	0	28	[8, 1, 15, 17, 19, 9, 16, 12, 18, 6, 2, 13, 14, 4, 10, 7, 11, 5, 20, 3]
	0,017467022	1388	0	47	[1, 6, 11, 14, 17, 5, 2, 19, 9, 4, 10, 7, 16, 8, 20, 15, 3, 18, 12, 13]
	0,037430048	1359	0	92	[9, 16, 6, 5, 3, 7, 17, 18, 8, 4, 2, 15, 14, 20, 11, 12, 1, 19, 10, 13]
	0,059888124	1357	0	150	[1, 15, 11, 7, 6, 4, 17, 10, 16, 14, 5, 19, 2, 8, 20, 9, 18, 12, 3, 13]
	0,412236691	1345	0	988	[15, 6, 5, 8, 12, 4, 16, 13, 10, 14, 1, 19, 9, 3, 11, 2, 17, 7, 18, 20]
	0,86888361	1332	0	2038	[15, 8, 4, 3, 6, 9, 19, 1, 17, 16, 2, 12, 14, 11, 5, 13, 18, 10, 7, 20]
	1,111414671	1323	0	2482	[15, 16, 3, 13, 14, 17, 19, 8, 5, 18, 1, 2, 9, 4, 11, 7, 12, 6, 10, 20]
	2,064161062	1320	0	4262	[6, 15, 8, 19, 13, 4, 2, 3, 16, 1, 11, 14, 5, 9, 18, 7, 17, 12, 10, 20]
	7,381271362	1297	0	13936	[15, 6, 3, 4, 9, 19, 14, 13, 11, 7, 1, 8, 17, 5, 18, 16, 12, 10, 2, 20]
	34,23532534	1296	1	58527	[15, 6, 9, 13, 11, 14, 4, 19, 5, 3, 7, 17, 8, 18, 2, 1, 16, 10, 20, 12]
	48,76180816	1295	2	78574	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 1, 2, 8, 19, 16, 10, 20, 12, 3]
	50,59889126	1294	2	81393	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 8, 16, 19, 1, 3, 2, 10, 20, 12]
	54,87441969	1293	3	88007	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 16, 1, 19, 2, 8, 10, 20, 12, 3]

	56,45000839	1288	4	90428	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 16, 1, 8, 2, 19, 10, 20, 12, 3]
	56,95906138	1287	4	91262	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 1, 2, 8, 16, 19, 10, 20, 12, 3]
15	0,000498772	1577	0	1	[20, 7, 8, 13, 14, 15, 5, 11, 9, 17, 12, 3, 16, 19, 18, 6, 4, 2, 10, 1]
	0,001496553	1536	0	4	[7, 2, 14, 19, 17, 16, 12, 5, 6, 11, 18, 3, 10, 8, 13, 9, 4, 15, 1, 20]
	0,001995802	1413	0	6	[17, 11, 8, 5, 14, 16, 3, 20, 7, 19, 1, 9, 15, 4, 2, 13, 6, 18, 12, 10]
	0,044433832	1407	0	115	[5, 16, 13, 7, 10, 1, 19, 14, 11, 8, 17, 12, 15, 2, 4, 3, 6, 9, 20, 18]
	0,051919937	1405	0	134	[7, 14, 8, 19, 1, 16, 17, 20, 3, 10, 13, 9, 4, 2, 15, 11, 5, 18, 12, 6]
	0,081365108	1377	0	210	[1, 16, 15, 14, 10, 4, 8, 19, 6, 9, 17, 5, 2, 11, 13, 12, 3, 7, 20, 18]
	0,095838308	1375	0	246	[1, 5, 17, 8, 4, 9, 10, 14, 11, 7, 3, 16, 13, 12, 19, 15, 6, 18, 2, 20]
	0,137261152	1344	0	348	[6, 8, 11, 14, 19, 15, 7, 2, 1, 18, 9, 3, 12, 4, 17, 13, 5, 16, 20, 10]
	0,277500391	1335	0	677	[1, 2, 15, 13, 11, 5, 16, 3, 6, 9, 14, 19, 4, 17, 12, 8, 7, 10, 18, 20]
	1,931904078	1333	0	4010	[15, 13, 1, 11, 19, 8, 7, 17, 16, 5, 3, 2, 4, 6, 9, 14, 10, 12, 18, 20]
	2,688498497	1332	0	5287	[15, 13, 14, 9, 17, 6, 16, 5, 1, 4, 7, 11, 18, 20, 2, 19, 10, 12, 8, 3]
	3,56586504	1326	0	6874	[15, 6, 7, 5, 12, 9, 17, 11, 4, 16, 1, 19, 13, 10, 14, 2, 8, 3, 18, 20]
	3,856825829	1309	0	7387	[15, 6, 13, 14, 11, 9, 16, 1, 2, 7, 3, 17, 4, 18, 12, 8, 19, 5, 10, 20]
	9,333157301	1305	0	17372	[15, 6, 9, 4, 16, 3, 19, 14, 8, 17, 10, 1, 5, 2, 13, 7, 11, 18, 20, 12]
	11,81104922	1304	0	21758	[15, 6, 9, 13, 14, 16, 5, 17, 2, 7, 4, 11, 19, 1, 8, 18, 10, 12, 3, 20]
	13,81680131	1297	0	25221	[15, 6, 9, 13, 14, 8, 2, 17, 11, 7, 5, 16, 3, 19, 4, 18, 1, 12, 10, 20]
	49,8597827	1291	2	81809	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 1, 17, 16, 2, 8, 19, 10, 20, 12, 3]
	52,83275318	1288	3	86277	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 16, 1, 8, 2, 19, 10, 20, 12, 3]
	54,78612161	1287	3	89321	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 16, 1, 19, 8, 2, 10, 20, 12, 3]
16	0,00049901	1529	0	1	[11, 20, 9, 10, 8, 4, 17, 7, 15, 14, 5, 3, 12, 6, 13, 19, 1, 18, 16, 2]
	0,001496792	1478	0	3	[15, 14, 11, 12, 18, 10, 9, 1, 3, 7, 8, 19, 2, 16, 4, 20, 5, 6, 17, 13]
	0,002994061	1475	0	8	[18, 5, 6, 3, 19, 14, 11, 17, 1, 8, 12, 2, 10, 9, 7, 13, 4, 16, 15, 20]
	0,003492594	1377	0	9	[1, 15, 6, 14, 3, 7, 17, 11, 9, 4, 10, 5, 13, 12, 20, 19, 2, 8, 16, 18]
	0,095338821	1356	0	246	[9, 5, 17, 8, 16, 2, 15, 3, 6, 14, 18, 19, 13, 12, 1, 20, 4, 11, 7, 10]
	0,571453333	1333	0	1354	[15, 6, 14, 4, 11, 13, 9, 12, 7, 8, 19, 2, 16, 1, 5, 10, 17, 18, 20, 3]
	0,61637044	1324	0	1458	[15, 13, 9, 3, 1, 8, 16, 11, 6, 19, 14, 5, 2, 4, 17, 12, 7, 10, 18, 20]
	5,361527205	1323	0	9884	[15, 9, 6, 2, 14, 7, 11, 17, 1, 5, 18, 12, 16, 13, 4, 8, 19, 10, 20, 3]
	6,509908915	1318	0	12055	[15, 6, 11, 17, 19, 13, 4, 9, 5, 7, 18, 16, 14, 10, 2, 1, 8, 3, 12, 20]
	6,838780165	1304	0	12683	[15, 6, 11, 14, 3, 2, 7, 9, 4, 8, 1, 19, 18, 17, 5, 12, 10, 16, 13, 20]
	8,687359095	1297	0	16175	[15, 6, 14, 17, 3, 5, 7, 8, 9, 11, 19, 1, 13, 12, 18, 16, 4, 2, 10, 20]
	23,69494653	1294	0	38735	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 1, 19, 18, 8, 2, 16, 10, 3, 20, 12]
	52,31022501	1293	2	83831	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 19, 1, 16, 8, 2, 10, 20, 12, 3]
	57,50206876	1287	4	92063	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 1, 19, 8, 2, 16, 10, 20, 12, 3]
17	0,000499487	1649	0	1	[16, 19, 9, 1, 8, 3, 12, 4, 18, 20, 5, 10, 14, 7, 13, 15, 6, 2, 11, 17]
	0,000998497	1456	0	2	[3, 20, 13, 17, 8, 19, 9, 18, 15, 11, 5, 12, 7, 6, 1, 14, 2, 4, 16, 10]
	0,002994776	1370	0	8	[8, 19, 15, 5, 16, 9, 6, 1, 17, 14, 11, 2, 13, 12, 18, 4, 10, 3, 20, 7]
	0,110295296	1316	0	276	[15, 19, 2, 4, 8, 3, 16, 9, 6, 1, 5, 7, 13, 11, 18, 12, 14, 17, 10, 20]
	6,870721579	1304	0	13106	[15, 6, 9, 2, 14, 7, 11, 5, 3, 1, 4, 16, 19, 17, 8, 13, 10, 18, 12, 20]
	10,29984355	1298	0	19393	[15, 6, 9, 14, 3, 5, 4, 12, 17, 13, 8, 19, 2, 18, 7, 1, 16, 11, 10, 20]
	10,65568185	1297	0	20031	[15, 6, 9, 14, 7, 3, 8, 16, 4, 1, 2, 11, 13, 19, 18, 17, 5, 12, 10, 20]
	43,19266677	1294	1	72599	[15, 6, 9, 13, 11, 14, 4, 3, 5, 18, 7, 17, 19, 1, 8, 2, 16, 10, 20, 12]
	53,61727786	1287	3	88876	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 1, 19, 16, 8, 2, 10, 20, 12, 3]
18	0,000516891	1500	0	1	[15, 9, 10, 8, 13, 16, 3, 1, 14, 6, 7, 18, 12, 19, 2, 5, 11, 4, 17, 20]
	0,001015663	1476	0	2	[14, 20, 6, 17, 15, 3, 5, 19, 16, 8, 9, 4, 2, 12, 11, 10, 7, 1, 13, 18]
	0,002013683	1446	0	5	[11, 4, 9, 8, 15, 19, 6, 17, 14, 1, 3, 20, 13, 16, 12, 7, 5, 18, 10, 2]
	0,003511429	1416	0	9	[4, 12, 17, 8, 9, 15, 7, 13, 3, 16, 11, 10, 5, 14, 6, 2, 19, 20, 1, 18]
	0,006505966	1396	0	19	[9, 6, 5, 1, 8, 12, 17, 19, 13, 18, 20, 4, 3, 2, 14, 16, 11, 15, 7, 10]
	0,042439461	1384	0	109	[15, 6, 14, 17, 1, 11, 18, 9, 19, 2, 4, 16, 20, 3, 10, 8, 5, 7, 13, 12]
	0,162216187	1376	0	405	[14, 6, 17, 7, 8, 11, 16, 1, 5, 18, 2, 3, 19, 4, 9, 13, 15, 12, 10, 20]
	0,206633568	1368	0	510	[6, 5, 9, 8, 13, 14, 7, 1, 15, 16, 11, 18, 12, 2, 4, 3, 20, 19, 17, 10]
	0,261033297	1363	0	631	[6, 5, 14, 4, 1, 2, 8, 19, 13, 16, 3, 9, 20, 17, 15, 11, 18, 12, 7, 10]
	0,429219723	1361	0	1014	[1, 17, 9, 11, 13, 4, 5, 16, 6, 15, 7, 18, 19, 8, 14, 10, 3, 12, 20, 2]
	0,490106344	1351	0	1156	[15, 14, 9, 5, 1, 18, 16, 17, 3, 4, 10, 8, 19, 2, 7, 11, 13, 6, 12, 20]
	0,502582788	1339	0	1180	[3, 1, 4, 6, 17, 12, 9, 5, 11, 15, 7, 13, 19, 16, 8, 18, 14, 2, 10, 20]
	0,997645378	1324	0	2295	[15, 4, 9, 1, 6, 7, 17, 11, 3, 14, 5, 8, 12, 19, 10, 13, 16, 2, 18, 20]
	4,86245656	1316	0	9265	[15, 6, 16, 5, 13, 4, 1, 3, 9, 14, 17, 11, 12, 7, 8, 19, 18, 2, 10, 20]
	5,765297174	1305	0	10927	[15, 1, 17, 11, 14, 8, 6, 9, 13, 16, 2, 3, 4, 19, 18, 12, 5, 10, 7, 20]
	7,586407661	1302	0	14376	[15, 6, 9, 13, 17, 4, 5, 1, 16, 18, 3, 19, 14, 12, 8, 11, 10, 2, 7, 20]
	10,63224363	1297	0	19858	[15, 6, 9, 7, 17, 19, 13, 14, 8, 1, 11, 12, 3, 5, 18, 16, 4, 2, 10, 20]
	23,30816722	1294	0	40733	[15, 6, 9, 13, 11, 14, 5, 19, 17, 7, 4, 2, 3, 1, 18, 8, 16, 10, 20, 12]
	63,9091537	1289	10	102666	[15, 6, 9, 13, 11, 14, 5, 19, 17, 7, 18, 4, 1, 16, 8, 2, 10, 20, 12, 3]
19	0,000498772	1496	0	1	[10, 19, 11, 4, 2, 15, 12, 17, 1, 6, 20, 18, 9, 5, 7, 3, 14, 8, 16, 13]
	0,000498772	1465	0	2	[16, 1, 13, 12, 4, 19, 7, 8, 3, 15, 6, 2, 17, 11, 20, 14, 10, 9, 5, 18]
	0,00099802	1393	0	3	[9, 3, 4, 16, 13, 6, 1, 11, 19, 14, 5, 7, 20, 18, 15, 12, 8, 17, 10, 2]
	0,023473263	1358	0	62	[4, 9, 17, 6, 1, 15, 13, 10, 3, 12, 5, 16, 7, 11, 18, 14, 19, 8, 20, 2]
	0,106318712	1340	0	272	[15, 7, 1, 11, 6, 3, 13, 12, 5, 14, 9, 4, 2, 16, 18, 19, 20, 8, 17, 10]
	1,377454996	1337	0	2818	[15, 6, 18, 3, 12, 7, 1, 17, 9, 11, 13, 4, 16, 2, 14, 5, 8, 19, 10, 20]
	1,929908752	1324	0	3856	[15, 3, 17, 1, 8, 11, 5, 14, 9, 4, 13, 2, 7, 6, 12, 16, 10, 19, 18, 20]
	3,979097605	1308	0	7425	[15, 6, 3, 14, 17, 19, 18, 1, 4, 16, 2, 8, 9, 11, 5, 13, 12, 10, 7, 20]
	4,97274971	1305	0	9236	[15, 6, 1, 8, 3, 17, 9, 19, 13, 14, 11, 7, 4, 16, 5, 10, 2, 18, 20, 12]
	6,522887468	1297	0	12213	[15, 6, 9, 13, 14, 4, 3, 17, 19, 11, 1, 5, 7, 2, 10, 16, 18, 12, 8, 20]

	20,10262823	1295	0	35348	[15, 6, 9, 13, 11, 14, 5, 18, 7, 16, 17, 1, 2, 19, 8, 3, 4, 10, 20, 12]
	22,36242557	1294	0	38880	[15, 6, 9, 13, 11, 14, 5, 4, 17, 16, 18, 3, 8, 19, 7, 1, 2, 10, 20, 12]
	53,96115518	1287	2	85117	[15, 6, 9, 13, 11, 14, 4, 5, 18, 7, 17, 1, 2, 8, 16, 19, 10, 20, 12, 3]
20	0,000499249	1561	0	1	[10, 6, 1, 8, 20, 15, 7, 16, 13, 5, 4, 2, 12, 9, 18, 19, 14, 17, 3, 11]
	0,000998259	1429	0	2	[15, 4, 14, 13, 16, 2, 19, 3, 1, 9, 11, 12, 17, 5, 8, 20, 7, 10, 6, 18]
	0,00397563	1414	0	10	[16, 9, 8, 19, 11, 14, 17, 1, 6, 4, 15, 13, 12, 2, 3, 20, 5, 7, 10, 18]
	0,010463476	1397	0	29	[13, 15, 16, 9, 6, 11, 3, 12, 19, 2, 14, 8, 4, 18, 5, 10, 7, 20, 17, 1]
	0,016950846	1386	0	45	[1, 9, 6, 19, 5, 7, 15, 16, 2, 13, 3, 14, 8, 12, 4, 18, 17, 11, 20, 10]
	0,042920351	1365	0	112	[14, 11, 1, 5, 15, 16, 6, 17, 7, 13, 9, 4, 10, 19, 18, 2, 12, 20, 3, 8]
	0,224065781	1358	0	540	[1, 13, 12, 19, 9, 8, 14, 11, 17, 16, 15, 5, 18, 3, 7, 6, 2, 4, 10, 20]
	0,476117611	1354	0	1133	[15, 14, 2, 1, 11, 13, 17, 19, 5, 4, 3, 10, 12, 6, 9, 18, 16, 8, 7, 20]
	0,481109142	1350	0	1144	[15, 13, 6, 11, 3, 14, 12, 5, 1, 19, 18, 17, 8, 9, 16, 7, 10, 4, 2, 20]
	0,548480034	1343	0	1287	[14, 17, 19, 6, 15, 11, 13, 9, 7, 18, 8, 5, 2, 4, 16, 20, 10, 12, 3, 1]
	1,254148245	1306	0	2769	[15, 9, 17, 2, 8, 3, 7, 11, 16, 13, 1, 6, 18, 19, 4, 14, 5, 12, 10, 20]
	11,21364331	1301	0	21268	[15, 6, 9, 13, 11, 17, 1, 7, 16, 19, 14, 3, 5, 4, 8, 18, 2, 12, 10, 20]
	15,56004238	1297	0	28800	[15, 6, 9, 13, 14, 11, 4, 16, 2, 19, 8, 3, 7, 5, 18, 12, 1, 17, 10, 20]
	49,4799552	1295	2	82236	[15, 6, 9, 13, 11, 14, 4, 5, 17, 7, 19, 1, 3, 8, 18, 16, 2, 10, 20, 12]
	54,4581964	1293	3	90145	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 19, 16, 8, 1, 2, 10, 20, 12, 3]
	55,76079035	1287	3	92238	[15, 6, 9, 13, 11, 14, 4, 5, 17, 18, 7, 1, 2, 16, 8, 19, 10, 20, 12, 3]

Załącznik 10: Przykładowy przebieg algorytmu MCTS dla zadania ta092

Czas	Funkcja celu	Węzeł	Iteracja	Uszeregowanie
0,02	12625	0	1	[77, 26, 70, 42, 171, 172, 156, 111, 165, 43, 166, 28, 55, 25, 93, 192, 177, 145, 48, 91, 85, 24, 95, 140, 75, 151, 197, 10, 23, 162, 181, 149, 194, 122, 30, 90, 170, 121, 39, 7, 198, 45, 50, 4, 157, 87, 128, 40, 130, 189, 150, 147, 163, 138, 106, 49, 37, 56, 84, 109, 132, 137, 199, 1, 80, 67, 184, 135, 133, 118, 57, 38, 76, 46, 98, 113, 86, 183, 73, 152, 188, 62, 35, 173, 83, 124, 74, 161, 31, 52, 36, 60, 27, 190, 72, 168, 79, 178, 127, 164, 180, 146, 59, 176, 154, 51, 63, 110, 15, 142, 105, 64, 158, 66, 101, 20, 125, 126, 141, 9, 53, 54, 200, 104, 34, 82, 94, 32, 41, 69, 160, 114, 16, 116, 78, 33, 47, 96, 144, 155, 2, 195, 100, 112, 129, 107, 174, 167, 108, 115, 29, 196, 6, 5, 81, 139, 169, 88, 89, 11, 58, 143, 92, 182, 102, 21, 22, 12, 117, 159, 18, 65, 134, 97, 186, 179, 136, 175, 187, 8, 119, 68, 120, 17, 61, 99, 193, 123, 153, 44, 191, 13, 14, 131, 19, 148, 103, 185, 3, 71]
0,03	12437	0	2	[137, 143, 7, 72, 84, 200, 11, 148, 113, 83, 170, 10, 157, 158, 73, 195, 168, 79, 3, 186, 150, 144, 100, 62, 181, 116, 27, 52, 128, 169, 192, 173, 153, 90, 4, 138, 125, 43, 164, 24, 13, 8, 95, 130, 117, 108, 115, 134, 161, 190, 104, 141, 2, 196, 86, 14, 57, 162, 71, 176, 189, 45, 97, 46, 34, 42, 146, 191, 75, 80, 12, 39, 48, 180, 123, 38, 199, 102, 114, 60, 152, 37, 171, 87, 96, 147, 129, 163, 89, 67, 15, 175, 105, 59, 159, 32, 101, 197, 82, 23, 35, 55, 182, 107, 109, 28, 110, 69, 66, 63, 124, 145, 194, 36, 149, 188, 47, 179, 51, 5, 92, 17, 88, 70, 58, 112, 65, 160, 120, 85, 49, 68, 61, 154, 185, 78, 183, 9, 64, 133, 53, 127, 41, 54, 26, 22, 184, 44, 31, 187, 50, 99, 126, 94, 155, 165, 18, 6, 77, 172, 122, 132, 76, 156, 1, 16, 139, 29, 136, 140, 151, 91, 25, 111, 118, 30, 178, 74, 174, 142, 93, 198, 177, 40, 21, 56, 167, 131, 19, 193, 106, 103, 135, 81, 20, 121, 98, 119, 166, 33]
0,05	12430	0	3	[156, 95, 98, 45, 188, 107, 112, 22, 58, 187, 62, 181, 80, 109, 49, 36, 152, 77, 69, 23, 115, 79, 10, 130, 102, 11, 51, 85, 189, 110, 30, 40, 116, 5, 19, 67, 44, 128, 129, 165, 197, 163, 200, 185, 27, 16, 28, 106, 17, 94, 56, 164, 139, 76, 169, 37, 137, 136, 190, 101, 103, 20, 134, 166, 81, 138, 192, 66, 144, 46, 25, 157, 54, 72, 87, 171, 186, 35, 117, 178, 149, 154, 50, 4, 21, 151, 78, 2, 90, 111, 61, 177, 183, 48, 105, 43, 161, 1, 104, 42, 158, 63, 176, 114, 159, 160, 52, 173, 97, 64, 132, 29, 174, 86, 91, 147, 99, 140, 122, 113, 196, 65, 142, 12, 96, 108, 182, 133, 70, 83, 55, 131, 172, 141, 32, 88, 15, 126, 118, 135, 39, 162, 84, 71, 41, 7, 8, 14, 127, 119, 9, 68, 34, 123, 13, 121, 89, 199, 60, 145, 184, 18, 26, 73, 170, 168, 195, 193, 148, 3, 93, 180, 198, 125, 124, 120, 75, 92, 33, 82, 194, 150, 100, 175, 155, 6, 167, 146, 74, 59, 143, 47, 24, 57, 191, 31, 179, 38, 153, 53]
0,08	12342	0	5	[89, 54, 34, 160, 80, 2, 111, 3, 36, 193, 186, 10, 116, 161, 68, 29, 86, 168, 153, 96, 79, 50, 98, 171, 5, 17, 118, 141, 112, 146, 176, 104, 57, 115, 31, 24, 90, 81, 35, 110, 61, 15, 198, 157, 113, 154, 69, 140, 185, 180, 21, 119, 20, 117, 123, 43, 143, 23, 58, 39, 144, 114, 129, 107, 11, 101, 73, 192, 138, 71, 51, 187, 188, 184, 178, 124, 97, 67, 22, 191, 175, 12, 19, 66, 169, 162, 151, 167, 52, 108, 53, 130, 106, 152, 181, 174, 38, 70, 32, 142, 87, 6, 164, 82, 74, 60, 7, 196, 88, 56, 135, 65, 147, 172, 149, 128, 1, 182, 48, 91, 94, 190, 197, 63, 121, 95, 18, 200, 127, 84, 120, 150, 170, 177, 40, 148, 103, 139, 122, 62, 30, 47, 16, 179, 156, 44, 158, 165, 72, 45, 8, 189, 13, 163, 64, 195, 194, 85, 9, 126, 145, 137, 166, 132, 99, 27, 83, 131, 55, 159, 14, 173, 33, 92, 42, 105, 4, 25, 183, 76, 49, 134, 100, 59, 46, 75, 93, 136, 26, 155, 133, 199, 77, 37, 102, 125, 28, 78, 109, 41]
0,11	11731	0	7	[20, 35, 65, 28, 44, 34, 6, 26, 150, 134, 31, 8, 104, 183, 180, 49, 118, 24, 127, 11, 114, 47, 13, 179, 168, 53, 95, 162, 110, 153, 40, 93, 48, 21, 200, 30, 119, 164, 97, 149, 120, 169, 191, 9, 145, 148, 133, 100, 182, 76, 17, 66, 80, 51, 12, 38, 15, 74, 138, 67, 83, 197, 16, 117, 192, 90, 27, 50, 128, 64, 154, 139, 58, 75, 79, 98, 155, 73, 156, 136, 63, 84, 116, 82, 158, 55, 159, 106, 112, 123, 141, 152, 108, 170, 78, 91, 143, 102, 199, 135, 190, 19, 85, 99, 109, 196, 129, 140, 166, 103, 43, 194, 18, 173, 62, 193, 132, 77, 176, 69, 92, 174, 113, 146, 131, 130, 126, 189, 36, 121, 172, 81, 54, 160, 195, 45, 101, 137, 59, 157, 181, 167, 5, 3, 88, 125, 165, 111, 22, 72, 89, 42, 71, 14, 57, 163, 4, 56, 29, 107, 7, 171, 23, 10, 52, 198, 60, 147, 68, 151, 124, 70, 61, 142, 41, 115, 175, 105, 178, 46, 122, 39, 184, 37, 188, 187, 185, 87, 2, 1, 144, 94, 96, 32, 161, 86, 186, 33, 25, 177]
3,69	11688	0	251	[78, 79, 133, 130, 152, 148, 135, 134, 24, 87, 154, 95, 75, 156, 160, 122, 57, 168, 60, 23, 179, 65, 25, 167, 144, 177, 43, 11, 124, 17, 121, 113, 125, 6, 72, 181, 54, 141, 146, 22, 145, 91, 153, 98, 77, 112, 20, 174, 136, 42, 41, 81, 138, 155, 137, 5, 89, 100, 59, 94, 180, 114, 182, 90, 64, 46, 4, 1, 127, 109, 103, 92, 99, 191, 151, 163, 40, 30, 193, 147, 15, 85, 105, 139, 170, 26, 195, 194, 173, 29, 35, 32, 27, 171, 88, 80, 192, 50, 197, 52, 44, 68, 53, 13, 9, 18, 33, 96, 164, 116, 187, 70, 62, 129, 8, 84, 159, 165, 38, 56, 106, 118, 86, 172, 55, 2, 108, 117, 66, 82, 58, 102, 161, 31, 115, 189, 200, 198, 149, 63, 76, 28, 111, 67, 119, 199, 12, 51, 175, 71, 107, 93, 123, 158, 49, 10, 19, 140, 74, 157, 184, 61, 34, 7, 37, 190, 47, 3, 150, 104, 131, 185, 21, 183, 128, 110, 83, 196, 14, 186, 39, 73, 169, 126, 143, 166, 101, 48, 132, 69, 45, 176, 97, 188, 162, 16, 142, 120, 178, 36]
6,49	11662	0	448	[1, 143, 65, 27, 20, 113, 97, 42, 40, 5, 23, 194, 8, 12, 73, 68, 38, 11, 16, 119, 149, 115, 44, 57, 102, 26, 56, 130, 155, 182, 63, 184, 176, 122, 4, 136, 167, 125, 120, 70, 175, 98, 150, 165, 129, 89, 166, 50, 25, 93, 87, 173, 140, 124, 79, 154, 72, 48, 33, 37, 147, 188, 45, 169, 92, 116, 128, 77, 137, 109, 138, 160, 103, 162, 146, 62, 21, 133, 187, 111, 157, 185, 196, 134, 190, 123, 152, 36, 151, 192, 2, 14, 6, 199, 78, 139, 118, 43, 141, 31, 86, 76, 82, 142, 34, 22, 91, 13, 186, 69, 110, 85, 191, 35, 168, 9, 58, 7, 64, 135, 66, 104, 24, 100, 170, 193, 30, 161, 121, 75, 108, 172, 41, 55, 80, 179, 28, 61, 156, 148, 29, 81, 195, 67, 83, 10, 181, 153, 94, 39, 49, 183, 17, 88, 159, 32, 107, 145, 127, 90, 177, 105, 117, 144, 60, 74, 51, 131, 99, 52, 84, 112, 106, 19, 101, 198, 197, 96, 163, 3, 53, 189, 180, 114, 18, 71, 132, 47, 200, 171, 178, 54, 158, 164, 174, 59, 95, 15, 46, 126]
18,50	11631	0	1293	[8, 111, 102, 194, 37, 148, 108, 56, 1, 65, 48, 31, 145, 99, 109, 169, 140, 86, 150, 4, 62, 143, 89, 128, 164, 138, 156, 133, 21, 63, 106, 186, 96, 168, 81, 47, 115, 147, 155, 165, 10, 52, 200, 195, 42, 87, 130, 196, 18, 172, 71, 184, 83, 54, 59, 124, 43, 153, 41, 16, 92, 117, 24, 55, 50, 72, 197, 182, 19, 142, 84, 151, 14, 114, 9, 90, 100, 135, 97, 185, 2, 51, 46, 166, 159, 158, 105, 141, 179, 88, 122, 167, 66, 20, 12, 22, 95, 157, 69, 49, 91, 35, 131, 82, 126, 139, 26, 15, 25, 28, 177, 27, 53, 36, 192, 77, 161, 33, 44, 76, 74, 103, 75, 38, 149, 193, 5, 162, 199, 94, 181, 175, 110, 174, 173, 39, 121, 113, 136, 78, 45, 67, 191, 127, 132, 13, 137, 101, 134, 61, 119, 190, 129, 163, 17, 6, 58, 23, 183, 198, 34, 170, 123, 178, 144, 30, 146, 187, 189, 85, 11, 120, 116, 171, 107, 160, 118, 112, 104, 176, 93, 40, 79, 57, 188, 152, 80, 7, 154, 29, 32, 125, 180, 73, 70, 3, 98, 64, 60, 68]
36,02	11584	0	2522	[19, 65, 174, 1, 49, 25, 130, 165, 178, 93, 192, 108, 32, 54, 58, 60, 177, 88, 52, 151, 127, 155, 77, 5, 162, 142, 85, 2, 12, 83, 89, 182, 39, 91, 35, 92, 115, 172, 197, 114, 44, 104, 168, 149, 150, 171, 101, 195, 56, 124, 62, 87, 106, 99, 186, 97, 167, 74, 64, 43, 198, 100, 176, 112, 191, 164, 42, 40, 28, 137, 76, 13, 184, 45, 159, 129, 123, 138, 179, 16, 194, 73, 18, 26, 173, 23, 46, 95, 17, 84, 175, 152, 20, 75, 98, 105, 139, 36, 140, 109, 136, 160, 122, 78, 102, 147, 200, 118, 166, 79, 3, 38, 22, 66, 153, 51, 68, 126, 10, 181, 199, 34, 170, 63, 158, 125, 180, 27, 117, 183, 169, 50, 134, 133, 41, 116, 71, 145, 57, 72, 190, 67, 128, 143, 154, 119, 90, 111, 33, 69, 9, 188, 6, 132, 82, 15, 81, 30, 103, 4, 110, 14, 96, 107, 21, 61, 120, 59, 31, 11, 135, 193, 121, 94, 29, 80, 189, 196, 53, 70, 37, 144, 86, 113, 185, 24, 131, 8, 7, 187, 47, 161, 141, 163, 55, 146, 157, 156, 48, 148]
55,03	11573	0	3860	[156, 171, 54, 81, 117, 146, 85, 27, 82, 43, 89, 97, 11, 169, 93, 144, 98, 174, 200, 194, 23, 175, 53, 13, 104, 114, 56, 134, 112, 57, 178, 103, 77, 20, 184, 129, 60, 122, 115, 163, 145, 139, 48, 137, 88, 124, 121, 192, 196, 19, 132, 135, 59, 177, 119, 127, 113, 70, 65, 1, 108, 87, 45, 107, 55, 15, 28, 187, 96, 83, 198, 22, 40, 68, 29, 16, 120, 61, 149, 105, 185, 21, 92, 50, 38, 142, 86, 106, 58, 72, 140, 164, 47, 190, 191, 91, 8, 150, 170, 74, 78, 130, 10, 111, 71, 4, 7, 116, 3, 173, 46, 110, 31, 84, 75, 94, 32, 181, 138, 168, 25, 133, 157, 12, 159, 64, 101, 154, 14, 90, 153, 26, 165, 66, 51, 37, 176, 30, 197, 34, 18, 100, 162, 44, 39, 95, 125, 160, 2, 67, 102, 79, 80, 189, 179, 63, 42, 52, 166, 161, 109, 62, 123, 141, 183, 152, 17, 9, 35, 199, 155, 143, 148, 186, 193, 6, 36, 76, 128, 69, 99, 158, 172, 136, 5, 131, 188, 118, 167, 33, 73, 180, 41, 182, 49, 147, 195, 126, 24, 151]
199,39	11560	0	13972	[7, 25, 73, 189, 72, 140, 16, 147, 112, 4, 29, 117, 58, 153, 44, 59, 141, 79, 182, 115, 128, 171, 41, 2, 187, 114, 87, 83, 179, 137, 157, 99, 155, 156, 194, 193, 178, 152, 46, 148, 127, 122, 13, 49, 190, 133, 195, 55, 177, 27, 74, 106, 146, 186, 165, 63, 62, 80, 91, 77, 184, 124, 78, 54, 135, 38, 5, 185, 22, 48, 82, 145, 88, 159, 65, 32, 131, 113, 84, 98, 136, 100, 158, 169, 103, 97, 123, 37, 66, 43, 12, 57, 120, 116, 160, 14, 56, 31, 70, 71, 134, 172, 61, 20, 176, 24, 161, 42, 166, 10, 11, 183, 111, 138, 28, 192, 26, 95, 181, 188, 162, 139, 104, 173, 86, 198, 199, 163, 154, 93, 64, 164, 105, 9, 21, 75, 129, 51, 107, 8, 191, 85, 53, 45, 52, 167, 196, 3, 96, 102, 144, 36, 108, 142, 151, 30, 110, 76, 132, 19, 180, 90, 67, 126, 35, 92, 170, 6, 121, 39, 150, 1, 15, 69, 174, 125, 33, 143, 68, 101, 60, 149, 50, 168, 34, 200, 40, 89, 175, 23, 17, 94, 18, 109, 197, 47, 130, 119, 81, 118]
575,14	11543	3	40387	[7, 25, 34, 92, 157, 21, 130, 147, 128, 140, 95, 42, 23, 66, 91, 155, 2, 119, 20, 48, 122, 165, 125, 105, 22, 68, 171, 75, 184, 97, 117, 197, 16, 200, 176, 133, 93, 112, 190, 145, 69, 177, 107, 83, 166, 45, 178, 77, 120, 152, 123, 193, 191, 156, 94, 153, 3, 106, 60, 8, 129, 164, 18, 179, 143, 154, 30, 39, 88, 98, 144, 160, 73, 195, 54, 15, 58, 31, 5, 103, 74, 180, 62, 78, 87, 137, 141, 40, 151, 182, 44, 189, 115, 101, 170, 64, 124, 86, 118, 111, 183, 174, 14, 131, 37, 150, 90, 172, 187, 139, 136, 173, 36, 47, 158, 121, 28, 110, 65, 61, 80, 194, 9, 19, 84, 29, 196, 17, 72, 46, 71, 1, 102, 10, 108, 63, 146, 149, 89, 114, 11, 99, 35, 175, 161, 67, 163, 49, 32, 81, 96, 13, 113, 38, 127, 132, 116, 192, 79, 59, 82, 43, 24, 148, 185, 6, 51, 70, 76, 138, 109, 41, 52, 162, 4, 168, 199, 53, 57, 159, 33, 56, 12, 134, 100, 167, 186, 50, 169, 188, 104, 181, 135, 27, 85, 142, 198, 26, 126, 55]
1273,41	11517	8	90901	[7, 25, 34, 92, 157, 181, 133, 182, 112, 134, 19, 95, 96, 85, 194, 168, 45, 146, 185, 198, 164, 158, 132, 16, 49, 82, 84, 60, 127, 152, 156, 124, 42, 163, 20, 147, 36, 114, 173, 5, 117, 37, 56, 167, 90, 26, 161, 113, 59, 166, 195, 88, 150, 149, 126, 169, 74, 141, 17, 40, 136, 97, 116, 47, 70, 111, 104, 65, 13, 170, 2, 28, 58, 125, 184, 122, 79, 80, 179, 109, 33, 1, 107, 22, 32, 183, 103, 78, 144, 81, 106, 102, 63, 189, 35, 51, 57, 3, 130, 137, 87, 160, 159, 135, 54, 61, 18, 46, 120, 153, 67, 73, 98, 69, 151, 100, 62, 8, 48, 172, 188, 192, 131, 14, 129, 86, 187, 110, 21, 99, 162, 178, 76, 29, 155, 10, 93, 31, 140, 119, 118, 50, 138, 91, 197, 64, 148, 24, 108, 143, 4, 171, 66, 89, 139, 53, 145, 186, 190, 38, 121, 200, 165, 193, 12, 101, 30, 15, 39, 72, 175, 94, 177, 6, 41, 105, 23, 27, 9, 115, 199, 123, 191, 43, 75, 68, 71, 52, 154, 142, 77, 11, 174, 176, 55, 180, 83, 44, 196, 128]
1333,17	11511	9	95355	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 152, 70, 74, 145, 37, 58, 30, 151, 90, 114, 141, 39, 75, 119, 65, 69, 137, 172, 78, 81, 129, 27, 122, 44, 32, 105, 108, 187, 146, 20, 6, 197, 48, 49, 1, 86, 56, 195, 164, 115, 3, 19, 28, 66, 99, 43, 29, 40, 52, 185, 153, 88, 15, 117, 23, 143, 192, 33, 71, 106, 83, 113, 124, 169, 55, 184, 60, 173, 180, 10, 121, 193, 31, 170, 84, 18, 102, 189, 198, 110, 125, 73, 100, 72, 9, 107, 63, 45, 163, 103, 178, 134, 38, 179, 191, 93, 89, 150, 8, 68, 186, 94, 174, 136, 54, 176, 142, 127, 95, 77, 130, 183, 159, 165, 166, 67, 42, 98, 80, 53, 87, 177, 62, 4, 50, 64, 109, 160, 199, 14, 12, 85, 118, 135, 111, 167, 126, 120, 2, 194, 149, 35, 144, 11, 41, 13, 97, 101, 24, 123, 168, 158, 46, 16, 76, 79, 91, 128, 96, 21, 22, 148, 61, 154, 171, 156, 116, 59, 155, 26, 175, 51, 200, 140, 57, 196, 138, 5, 17, 162, 139, 147, 131, 188, 47, 132, 82, 190, 36]
1715,00	11504	16	124584	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 5, 109, 122, 184, 50, 57, 103, 173, 155, 120, 96, 47, 119, 12, 39, 146, 75, 171, 106, 130, 77, 68, 2, 15, 118, 66, 197, 134, 142, 14, 27, 29, 196, 19, 71, 60, 46, 90, 33, 156, 38, 91, 89, 100, 143, 105, 174, 36, 32, 137, 23, 163, 168, 144, 117, 115, 198, 49, 192, 61, 101, 147, 150, 4, 132, 179, 9, 3, 166, 22, 20, 199, 110, 129, 17, 52, 86, 178, 56, 159, 72, 149, 188, 10, 87, 152, 138, 102, 123, 18, 167, 13, 85, 111, 35, 194, 164, 200, 195, 83, 131, 30, 84, 42, 59, 175, 28, 65, 37, 153, 140, 145, 53, 190, 185, 121, 158, 127, 94, 76, 51, 98, 191, 80, 62, 116, 48, 95, 97, 79, 11, 170, 124, 81, 165, 31, 43, 73, 186, 67, 113, 69, 21, 162, 41, 26, 180, 108, 128, 176, 193, 78, 107, 16, 169, 135, 125, 6, 114, 70, 54, 151, 141, 99, 177, 8, 63, 183, 160, 74, 58, 139, 189, 136, 40, 187, 126, 44, 88, 55, 172, 82, 154]
1757,68	11454	17	127968	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 180, 138, 58, 155, 13, 51, 154, 137, 49, 189, 9, 163, 146, 198, 168, 84, 48, 90, 87, 50, 94, 27, 5, 57, 16, 115, 80, 170, 36, 75, 20, 91, 19, 11, 95, 105, 150, 73, 123, 178, 152, 77, 31, 113, 32, 200, 132, 173, 89, 171, 159, 139, 56, 47, 149, 134, 169, 98, 166, 100, 183, 65, 151, 174, 192, 124, 52, 29, 176, 106, 2, 120, 186, 14, 127, 88, 102, 8, 118, 156, 165, 162, 108, 107, 188, 18, 167, 145, 22, 164, 125, 193, 35, 71, 81, 53, 153, 17, 190, 21, 195, 70, 15, 131, 158, 41, 55, 67, 82, 12, 129, 144, 3, 142, 128, 197, 38, 23, 83, 140, 135, 175, 130, 54, 99, 110, 78, 44, 147, 72, 136, 185, 60, 68, 143, 86, 30, 172, 66, 184, 28, 114, 191, 101, 39, 43, 59, 74, 179, 96, 26, 97, 187, 40, 199, 116, 122, 111, 160, 141, 10, 46, 119, 42, 69, 4, 37, 85, 61, 126, 33, 62, 76, 196, 117, 79, 194, 63, 6, 109, 103, 121, 177]
2068,40	11434	39	153839	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 180, 138, 184, 3, 57, 169, 33, 156, 185, 98, 88, 179, 127, 113, 35, 15, 83, 150, 108, 42, 61, 160, 100, 166, 44, 102, 117, 82, 30, 131, 163, 69, 54, 78, 41, 176, 89, 173, 63, 171, 191, 143, 99, 178, 74, 125, 189, 101, 6, 58, 106, 193, 165, 16, 72, 158, 186, 76, 65, 194, 62, 85, 43, 68, 190, 11, 32, 145, 37, 109, 91, 84, 60, 21, 167, 71, 174, 164, 23, 132, 123, 196, 152, 200, 146, 195, 47, 141, 13, 9, 120, 172, 73, 22, 2, 59, 53, 26, 153, 168, 130, 14, 142, 19, 151, 70, 155, 197, 20, 107, 96, 38, 97, 56, 110, 198, 119, 144, 126, 46, 39, 177, 4, 170, 55, 17, 12, 103, 116, 75, 87, 118, 90, 105, 8, 129, 40, 121, 124, 95, 139, 114, 135, 137, 115, 140, 27, 67, 49, 128, 18, 122, 48, 50, 111, 154, 136, 192, 188, 51, 81, 199, 77, 162, 5, 28, 94, 29, 36, 183, 134, 86, 31, 52, 80, 147, 159, 66, 10, 175, 149, 187, 79]
2117,65	11405	55	158859	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 180, 138, 184, 3, 57, 169, 33, 156, 185, 98, 88, 179, 127, 113, 35, 15, 83, 150, 108, 42, 61, 160, 100, 166, 158, 16, 142, 91, 29, 74, 152, 117, 4, 162, 68, 194, 120, 187, 118, 55, 183, 66, 163, 107, 95, 65, 131, 168, 62, 134, 20, 164, 140, 116, 145, 167, 8, 19, 73, 171, 48, 200, 172, 103, 54, 196, 135, 137, 149, 32, 199, 96, 71, 63, 193, 52, 186, 27, 136, 18, 13, 106, 22, 101, 122, 46, 130, 2, 189, 141, 26, 58, 129, 36, 39, 165, 89, 99, 49, 43, 144, 110, 176, 60, 41, 40, 175, 174, 72, 82, 78, 155, 85, 123, 190, 154, 191, 38, 198, 97, 121, 128, 153, 173, 47, 67, 105, 125, 90, 114, 159, 81, 195, 94, 51, 12, 146, 59, 132, 69, 77, 14, 177, 197, 192, 11, 75, 9, 56, 188, 143, 21, 37, 87, 53, 80, 28, 84, 79, 31, 10, 50, 170, 44, 76, 115, 5, 126, 86, 124, 119, 6, 147, 17, 102, 30, 178, 109, 151, 70, 23, 139, 111]
2162,45	11354	70	163945	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 180, 138, 184, 3, 57, 169, 33, 156, 185, 98, 88, 179, 127, 113, 35, 15, 83, 150, 108, 42, 61, 160, 100, 166, 158, 16, 142, 91, 29, 74, 152, 117, 4, 162, 68, 194, 120, 187, 118, 55, 198, 72, 66, 30, 188, 12, 165, 101, 78, 153, 14, 146, 110, 56, 50, 18, 37, 80, 26, 173, 151, 9, 40, 107, 59, 43, 2, 94, 54, 11, 130, 197, 176, 8, 114, 28, 190, 135, 71, 63, 86, 140, 175, 46, 124, 170, 73, 183, 155, 82, 128, 20, 131, 163, 103, 44, 109, 5, 191, 125, 58, 147, 38, 41, 115, 97, 192, 178, 186, 84, 171, 27, 69, 31, 164, 76, 36, 122, 21, 90, 123, 195, 23, 99, 134, 53, 136, 22, 119, 70, 6, 89, 145, 102, 48, 47, 32, 159, 95, 199, 144, 105, 129, 67, 77, 177, 149, 39, 52, 51, 13, 167, 143, 87, 19, 168, 139, 60, 174, 79, 132, 85, 196, 81, 126, 111, 65, 172, 141, 193, 75, 10, 200, 62, 121, 96, 137, 116, 49, 154, 106, 189, 17]
2190,48	11334	80	167472	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 180, 138, 184, 3, 57, 169, 33, 156, 185, 98, 88, 179, 127, 113, 35, 15, 83, 150, 108, 42, 61, 160, 100, 166, 158, 16, 142, 91, 29, 74, 152, 117, 4, 162, 68, 194, 120, 187, 118, 55, 198, 72, 66, 30, 188, 12, 165, 101, 78, 153, 14, 146, 110, 56, 50, 49, 145, 171, 22, 168, 144, 31, 128, 122, 67, 85, 82, 129, 89, 177, 41, 81, 195, 119, 73, 2, 5, 99, 154, 124, 38, 190, 155, 175, 102, 95, 11, 167, 139, 75, 164, 126, 37, 192, 131, 151, 20, 140, 191, 197, 9, 19, 135, 6, 196, 97, 28, 178, 46, 163, 147, 39, 21, 77, 69, 65, 13, 121, 183, 114, 54, 170, 137, 173, 115, 143, 125, 90, 174, 63, 149, 17, 172, 60, 189, 84, 80, 71, 136, 116, 94, 36, 96, 186, 199, 130, 86, 47, 123, 44, 62, 51, 200, 27, 105, 32, 176, 53, 107, 111, 52, 48, 18, 109, 103, 8, 70, 58, 43, 59, 10, 141, 76, 40, 26, 106, 159, 132, 23, 193, 87, 79]
2213,03	11306	87	170535	[7, 25, 34, 92, 157, 181, 133, 182, 112, 104, 161, 45, 1, 148, 64, 24, 93, 180, 138, 184, 3, 57, 169, 33, 156, 185, 98, 88, 179, 127, 113, 35, 15, 83, 150, 108, 42, 61, 160, 100, 166, 158, 16, 142, 91, 29, 74, 152, 117, 4, 162, 68, 194, 120, 187, 118, 55, 198, 72, 66, 30, 188, 12, 165, 101, 78, 153, 14, 146, 110, 56, 50, 49, 145, 171, 22, 168, 144, 31, 128, 122, 5, 124, 106, 114, 80, 54, 86, 191, 107, 154, 17, 81, 13, 84, 173, 172, 126, 38, 79, 177, 137, 96, 130, 97, 62, 193, 28, 77, 82, 102, 75, 59, 19, 9, 140, 85, 21, 143, 175, 8, 41, 136, 109, 26, 167, 89, 125, 39, 134, 69, 164, 115, 199, 6, 67, 48, 95, 116, 58, 103, 2, 10, 73, 87, 11, 192, 99, 36, 200, 129, 186, 44, 90, 43, 105, 132, 52, 147, 163, 151, 170, 111, 53, 65, 149, 71, 183, 121, 18, 174, 123, 60, 195, 139, 46, 159, 155, 190, 189, 51, 20, 27, 197, 32, 40, 196, 76, 131, 176, 37, 70, 94, 119, 135, 63, 178, 47, 23, 141]

Załącznik 11: Kod programu realizujący moduł użytkowy – eksport danych do Excel oraz rysowanie harmonogramu Gantta.

```
import plotly.express as px
import pandas as pd
import numpy as np
import datetime

def plotGantt(nazwa,start_date,  NWR, NWZ, NPR, NPZ):
    NWR = np.array(NWR)
    NWZ = np.array(NWZ)
    NPR = np.array(NPR)
    NPZ = np.array(NPZ)

    n, m = NWR.shape

    name_list = []
    color = []
    for i in range(n):
        for j in range(m):
            name_list.append(f"{nazwa[i]}B{j + 1} NW")
            name_list.append(f"{nazwa[i]}B{j + 1} NP")
            color.append(f"B{j + 1}1")
            color.append(f"B{j + 1}2")

    NWR = NWR.flatten().tolist()
    NWZ = NWZ.flatten().tolist()
    NPR = NPR.flatten().tolist()
    NPZ = NPZ.flatten().tolist()

    R = []

    for x, y in zip(NWR, NPR):
        R.append(x)
        R.append(y)

    Z = []

    for x, y in zip(NWZ, NPZ):
        Z.append(x)
        Z.append(y)
    NWt = []
    NPt = []

    for i in range(len(R)):
        NWt.append((start_date + datetime.timedelta(days=R[i])).strftime("%Y-%m-%d"))
        NPt.append((start_date + datetime.timedelta(days=Z[i])).strftime("%Y-%m-%d"))

    data = {'Nazwa': name_list,
            'Data rozpoczęcia': NWt,
            'Data zakończenia': NPt,
            'Color': color}
    df = pd.DataFrame(data)

    fig = px.timeline(df, x_start='Data rozpoczęcia', x_end='Data zakończenia', y='Nazwa', color='Color',
                      category_orders={"Nazwa": name_list})
    fig.show()

def exportToExcel(nazwa,start_date,  NWR, NWZ, NPR, NPZ):

    NWR = np.array(NWR)
    NWZ = np.array(NWZ)
    NPR = np.array(NPR)
    NPZ = np.array(NPZ)

    start_date = datetime.date(2023, 1, 1)

    n, m = NWR.shape

    name_list = []
    color = []
    for i in range(n):
        for j in range(m):
            name_list.append(f"{nazwa[i]}B{j + 1}")

    NWR = NWR.flatten().tolist()
    NWZ = NWZ.flatten().tolist()
    NPR = NPR.flatten().tolist()
    NPZ = NPZ.flatten().tolist()

    NWRt = []
```

```
    NWZt = []
    NPRt = []
    NPZt = []

    for i in range(len(NWR)):
        NWRt.append((start_date + datetime.timedelta(days=NWR[i])).strftime("%Y-%m-%d"))
        NWZt.append((start_date + datetime.timedelta(days=NWZ[i])).strftime("%Y-%m-%d"))
        NPRt.append((start_date + datetime.timedelta(days=NPR[i])).strftime("%Y-%m-%d"))
        NPZt.append((start_date + datetime.timedelta(days=NPZ[i])).strftime("%Y-%m-%d"))

    df = pd.DataFrame({'Nazwa': name_list,
                       'Najwcześniejsze rozpoczęcie': NWRt,
                       'Najwcześniejsze zakończenie': NWZt,
                       'Najpóźniejsze rozpoczęcie': NPRt,
                       'Najpóźniejsze zakończenie': NPZt})

    df.to_excel('harmonogram.xlsx', sheet_name='Arkusz1', index=False)

if __name__ == "__main__":
    name = "C4"
    NWR = [[0, 7, 16, 22],
           [7, 16, 22, 29],
           [16, 26, 33, 40]]
    NWZ = [[7, 15, 22, 29],
           [16, 20, 29, 38],
           [26, 33, 40, 44]]
    NPR = [[0, 10, 18, 24],
           [7, 20, 24, 31],
           [16, 26, 33, 40]]
    NPZ = [[7, 18, 24, 31],
           [16, 24, 31, 40],
           [26, 33, 40, 44]]

    nazwa = ["O1", "O2", "O3"]

    start_date = datetime.date(2023, 1, 1)

    plotGantt(nazwa, start_date, NWR, NWZ, NPR, NPZ)
    exportToExcel(nazwa, start_date, NWR, NWZ, NPR, NPZ)
```

Załącznik 12: Przykładowy przebieg algorytmu MCTS dla zadania ta041 z uwzględnieniem ograniczeń kolejności wykonania obiektów

Czas	Funkcja celu	Węzeł	Iteracja	Uszeregowanie
0,001026392	3590.0	0	1	[34, 3, 42, 35, 39, 36, 33, 46, 20, 18, 1, 19, 22, 38, 7, 6, 30, 40, 48, 44, 50, 24, 28, 49, 13, 15, 21, 37, 31, 41, 14, 11, 8, 17, 4, 5, 25, 29, 26, 45, 10, 9, 16, 2, 27, 32, 43, 12, 47, 23]
181,2162185	3502.0	1	208	[34, 3, 37, 7, 6, 15, 29, 24, 46, 22, 27, 28, 49, 33, 19, 1, 18, 45, 21, 13, 35, 14, 11, 42, 47, 20, 8, 36, 31, 2, 40, 10, 41, 30, 38, 39, 43, 17, 25, 44, 4, 5, 26, 50, 16, 23, 9, 12, 32, 48]
184,1883307	3478.0	1	1985	[34, 3, 25, 29, 38, 37, 24, 43, 48, 7, 6, 22, 18, 19, 1, 8, 49, 21, 42, 26, 10, 50, 20, 46, 35, 41, 14, 11, 13, 47, 44, 36, 2, 30, 45, 31, 15, 28, 33, 17, 4, 5, 27, 40, 32, 9, 39, 16, 12, 23]
185,6805809	3477.0	1	2876	[34, 3, 24, 25, 28, 21, 15, 12, 18, 19, 1, 26, 16, 6, 7, 35, 38, 33, 20, 4, 5, 10, 39, 17, 43, 2, 32, 30, 36, 50, 41, 40, 13, 27, 47, 29, 49, 46, 42, 14, 11, 23, 8, 45, 37, 31, 22, 9, 48, 44]
188,0580404	3391.0	1	4288	[34, 3, 27, 13, 36, 20, 49, 15, 19, 18, 1, 7, 6, 42, 21, 40, 44, 31, 43, 10, 33, 2, 16, 37, 39, 45, 32, 14, 11, 28, 8, 24, 48, 41, 30, 12, 29, 50, 47, 38, 4, 5, 46, 35, 17, 22, 9, 26, 23, 25]
227,0515811	3382.0	1	27462	[34, 3, 19, 18, 1, 29, 49, 23, 32, 40, 38, 15, 13, 26, 31, 28, 39, 10, 36, 33, 12, 8, 16, 4, 5, 17, 6, 7, 21, 41, 20, 2, 43, 48, 14, 11, 35, 9, 42, 47, 24, 37, 27, 22, 30, 44, 46, 50, 45, 25]
420,7246146	3353.0	2	144619	[34, 3, 18, 19, 1, 20, 43, 28, 49, 36, 12, 24, 22, 10, 2, 44, 33, 16, 21, 32, 31, 6, 7, 45, 14, 11, 42, 13, 50, 30, 41, 17, 8, 47, 29, 15, 40, 9, 35, 46, 27, 23, 39, 37, 38, 4, 5, 26, 25, 48]
592,7569382	3334.0	3	251204	[34, 3, 18, 19, 1, 20, 15, 49, 22, 44, 13, 36, 43, 47, 21, 27, 17, 16, 12, 26, 4, 5, 42, 37, 30, 2, 33, 28, 46, 41, 6, 7, 40, 35, 50, 10, 29, 14, 11, 39, 32, 38, 8, 23, 31, 25, 9, 48, 24, 45]
645,1603372	3330.0	4	283877	[34, 3, 18, 19, 1, 20, 33, 15, 43, 49, 16, 36, 13, 31, 6, 7, 2, 38, 41, 4, 5, 37, 27, 32, 45, 44, 14, 11, 22, 23, 40, 35, 42, 28, 47, 25, 8, 12, 10, 9, 46, 50, 26, 29, 48, 17, 21, 30, 39, 24]
912,6449738	3329.0	6	451246	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 31, 38, 16, 8, 17, 7, 6, 47, 14, 11, 30, 4, 5, 42, 13, 12, 37, 29, 2, 45, 44, 46, 9, 25, 22, 28, 27, 40, 39, 10, 32, 23, 36, 26, 50, 24, 41, 48, 21, 35]
949,4639685	3318.0	7	474956	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 38, 29, 2, 32, 30, 10, 40, 22, 8, 24, 46, 27, 36, 12, 42, 44, 25, 13, 16, 28, 39, 17, 14, 11, 31, 21, 6, 7, 47, 45, 4, 5, 48, 50, 23, 9, 37, 35, 41, 26]
1041,392968	3275.0	8	535353	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 14, 11, 41, 47, 40, 36, 4, 5, 35, 13, 42, 31, 2, 30, 10, 45, 29, 24, 46, 28, 44, 22, 50, 27, 17, 23, 16, 8, 25, 32, 21, 12, 9, 48, 39, 26]
1411,635411	3267.0	14	803156	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 46, 14, 11, 32, 10, 12, 50, 31, 30, 28, 13, 29, 16, 36, 39, 17, 42, 22, 44, 24, 40, 21, 8, 4, 5, 27, 2, 45, 47, 35, 23, 9, 26, 48, 25]
1461,672374	3260.0	15	843602	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 28, 30, 12, 29, 44, 35, 24, 8, 14, 11, 17, 50, 27, 26, 40, 23, 16, 47, 4, 5, 42, 25, 31, 9, 2, 48, 22, 32, 39, 10, 45]
1543,046514	3250.0	18	912619	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 50, 31, 8, 2, 26, 12, 23, 9, 14, 11, 42, 24, 22, 29, 4, 5, 10, 16, 17, 35, 28, 27, 48, 47, 30, 32, 45, 40, 25, 39]
1572,154099	3239.0	19	938843	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 29, 40, 26, 14, 11, 8, 31, 32, 47, 50, 27, 24, 35, 12, 2, 4, 5, 10, 42, 22, 17, 48, 23, 9, 25, 28, 16, 30, 39, 45]
1609,254924	3238.0	20	973894	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 32, 35, 23, 28, 31, 10, 30, 16, 14, 11, 17, 8, 9, 4, 5, 26, 2, 48, 27, 22, 12, 29, 40, 47, 50, 45, 39, 24]
1649,452618	3214.0	22	1014378	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 30, 4, 5, 31, 28, 22, 27, 17, 2, 50, 32, 10, 12, 14, 11, 23, 24, 16, 40, 35, 29, 8, 47, 9, 48, 45, 26, 39]
1727,533052	3212.0	29	1105033	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 30, 4, 5, 31, 28, 14, 11, 17, 23, 40, 16, 8, 12, 9, 32, 29, 35, 27, 39, 50, 2, 22, 47, 26, 24, 48, 10, 45]
1739,774064	3198.0	30	1121050	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 30, 4, 5, 31, 28, 14, 11, 17, 22, 29, 40, 23, 10, 50, 16, 2, 27, 39, 35, 32, 8, 9, 47, 12, 26, 48, 45, 24]
1761,840915	3193.0	34	1152534	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 30, 4, 5, 31, 28, 14, 11, 17, 22, 29, 40, 23, 50, 32, 35, 26, 16, 10, 8, 9, 47, 2, 27, 24, 12, 45, 48, 39]
1761,903723	3183.0	34	1152633	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 30, 4, 5, 31, 28, 14, 11, 17, 22, 29, 40, 23, 8, 9, 32, 2, 27, 35, 26, 16, 12, 50, 48, 47, 10, 24, 45, 39]
1774,171308	3182.0	37	1172053	[34, 3, 18, 19, 1, 20, 33, 43, 49, 15, 37, 6, 7, 38, 41, 21, 36, 46, 13, 44, 25, 42, 30, 4, 5, 31, 28, 14, 11, 17, 22, 29, 40, 23, 8, 9, 32, 35, 27, 47, 2, 16, 50, 24, 10, 12, 26, 48, 45, 39]

Załącznik 13: Kod programu w języku Python realizujący moduł użytkowy

```
import plotly.express as px
import pandas as pd
import numpy as np
import datetime

def plotGantt(nazwa,start_date,  NWR, NWZ, NPR, NPZ, brigades=[]):


    NWR = np.array(NWR)
    NWZ = np.array(NWZ)
    NPR = np.array(NPR)
    NPZ = np.array(NPZ)

    n, m = NWR.shape

    if not brigades:
        brigades=list(range(m))

    name_list = []
    color = []
    for i in range(n):
        for j in range(m):
            if j in brigades:
                name_list.append(f"{nazwa[i]}B{j + 1} NW")
                name_list.append(f"{nazwa[i]}B{j + 1} NP")
                color.append(f"B{j + 1}1")
                color.append(f"B{j + 1}2")
    print(NWR)
    print(NWR[:,brigades].flatten().tolist())

    NWR = NWR[:,brigades].flatten().tolist()
    NWZ = NWZ[:,brigades].flatten().tolist()
    NPR = NPR[:,brigades].flatten().tolist()
    NPZ = NPZ[:,brigades].flatten().tolist()

    R = []

    for x, y in zip(NWR, NPR):
        R.append(x)
        R.append(y)

    Z = []

    for x, y in zip(NWZ, NPZ):
        Z.append(x)
        Z.append(y)
    NWt = []
    NPt = []

    for i in range(len(R)):
        NWt.append((start_date + datetime.timedelta(days=R[i])).strftime("%Y-%m-%d"))
        NPt.append((start_date + datetime.timedelta(days=Z[i])).strftime("%Y-%m-%d"))

    data = {'Nazwa': name_list,
            'Data rozpoczęcia': NWt,
            'Data zakończenia': NPt,
            'Color': color}
    df = pd.DataFrame(data)

    fig = px.timeline(df, x_start='Data rozpoczęcia', x_end='Data zakończenia', y='Nazwa', color='Color',
                      category_orders={"Nazwa": name_list})
    fig.update_xaxes(tickformat="%d-%m-%Y")
    fig.show()

def exportToExcel(nazwa,nazwa_pliku, start_date, NWR, NWZ, NPR, NPZ):

    NWR = np.array(NWR)
    NWZ = np.array(NWZ)
    NPR = np.array(NPR)
    NPZ = np.array(NPZ)

    n, m = NWR.shape

    name_list = []
    for i in range(n):
        for j in range(m):
            name_list.append(f"{nazwa[i]}B{j + 1}")

    NWR = NWR.flatten().tolist()
```

```
    NWZ = NWZ.flatten().tolist()
    NPR = NPR.flatten().tolist()
    NPZ = NPZ.flatten().tolist()

    NWRt = []
    NWZt = []
    NPRt = []
    NPZt = []

    for i in range(len(NWR)):
        NWRt.append((start_date + datetime.timedelta(days=NWR[i])).strftime("%Y-%m-%d"))
        NWZt.append((start_date + datetime.timedelta(days=NWZ[i])).strftime("%Y-%m-%d"))
        NPRt.append((start_date + datetime.timedelta(days=NPR[i])).strftime("%Y-%m-%d"))
        NPZt.append((start_date + datetime.timedelta(days=NPZ[i])).strftime("%Y-%m-%d"))

    df = pd.DataFrame({'Nazwa': name_list,
                       'Najwcześniejsze rozpoczęcie': NWRt,
                       'Najwcześniejsze zakończenie': NWZt,
                       'Najpóźniejsze rozpoczęcie': NPRt,
                       'Najpóźniejsze zakończenie': NPZt})

    df.to_excel(f'{nazwa_pliku}.xlsx', sheet_name='Arkusz1', index=False)

def genNazwa(list):
    return [f"O{i}" for i in list]
```

Załącznik 14: Kod programu w języku Python realizujący przykład 1

```
from PS import PS
from prj import *

'''Stałe'''
L1 = 1e4
L2 = 1e6
L3 = 1e8
L4 = 1e10
L5 = 1e12
L6 = 1e14

'''Zadanie praktyczne 1'''
name = "P1"

kgr = [[38.4, 70.5, 51.9, 62.7, 33.1],
      [33.2, 68.1, 68.7, 69.2, 35.5],
      [39.2, 75.2, 68.3, 68.7, 38.4],
      [38.4, 77.8, 54.1, 64.1, 33.3],
      [35.5, 64.5, 57.5, 68.3, 31.1],
      [32.5, 68.1, 54.9, 63.9, 32.5],
      [39.6, 71.5, 65.9, 70.0, 39.5],
      [38.4, 79.2, 63.2, 61.2, 34.3],
      [39.0, 71.8, 56.5, 64.8, 36.0],
      [39.2, 61.8, 60.6, 62.1, 34.0],
      [30.4, 69.1, 50.9, 67.5, 35.5],
      [38.2, 63.8, 62.5, 68.5, 36.7],
      [32.3, 66.1, 55.8, 61.0, 37.9],
      [30.2, 65.2, 58.0, 69.8, 34.6],
      [35.6, 62.4, 58.5, 61.4, 34.7],
      [36.6, 72.0, 56.2, 64.6, 33.1],
      [36.3, 78.6, 69.3, 63.0, 33.9],
      [30.6, 75.9, 67.1, 62.5, 33.4],
      [37.0, 67.7, 60.7, 61.0, 38.3],
      [31.1, 70.2, 51.9, 60.5, 37.1]]
tgr = [[11.0, 43.0, 39.0, 44.0, 42.0],
      [13.0, 47.0, 35.0, 36.0, 31.0],
      [12.0, 48.0, 32.0, 30.0, 37.0],
      [17.0, 46.0, 37.0, 43.0, 31.0],
      [14.0, 49.0, 39.0, 31.0, 41.0],
      [12.0, 48.0, 33.0, 43.0, 39.0],
      [19.0, 44.0, 32.0, 31.0, 56.0],
      [11.0, 47.0, 33.0, 34.0, 31.0],
      [17.0, 46.0, 34.0, 38.0, 30.0],
      [12.0, 40.0, 40.0, 41.0, 40.0],
      [18.0, 45.0, 31.0, 42.0, 32.0],
      [12.0, 43.0, 37.0, 45.0, 32.0],
      [15.0, 46.0, 38.0, 42.0, 37.0],
      [16.0, 44.0, 36.0, 36.0, 58.0],
      [13.0, 44.0, 31.0, 37.0, 36.0],
      [16.0, 45.0, 35.0, 43.0, 31.0],
      [14.0, 44.0, 31.0, 45.0, 48.0],
      [12.0, 46.0, 34.0, 31.0, 58.0],
      [11.0, 44.0, 31.0, 31.0, 31.0],
      [19.0, 46.0, 36.0, 48.0, 51.0]]
t = [[[1, 1, 2], [3, 1, 2], [1, 3, 1], [3, 1, 2], [2, 2, 2]], [[3, 2, 1], [2, 1, 3], [2, 1, 1], [3, 2, 2], [3, 1, 1]],
    [[2, 3, 3], [2, 2, 1], [2, 2, 2], [3, 1, 3], [3, 1, 3]], [[2, 1, 3], [3, 3, 3], [1, 3, 3], [2, 1, 1], [3, 1, 1]],
    [[3, 1, 1], [1, 1, 3], [1, 2, 2], [2, 3, 1], [3, 1, 1]], [[1, 3, 2], [3, 3, 1], [3, 2, 3], [3, 2, 1], [3, 3, 3]],
    [[3, 2, 3], [3, 2, 2], [1, 1, 1], [2, 1, 2], [3, 1, 3]], [[3, 3, 2], [1, 1, 2], [2, 3, 3], [1, 3, 2], [1, 3, 3]],
    [[3, 3, 3], [3, 3, 2], [2, 1, 3], [3, 3, 3], [3, 2, 3]], [[3, 1, 1], [3, 2, 1], [2, 3, 2], [3, 2, 2], [2, 2, 3]],
    [[2, 1, 1], [3, 2, 2], [3, 2, 2], [3, 1, 3], [1, 3, 2]], [[1, 3, 3], [2, 1, 1], [2, 1, 2], [3, 3, 3], [1, 1, 3]],
    [[3, 1, 1], [2, 3, 3], [2, 3, 1], [2, 2, 2], [3, 2, 1]], [[2, 3, 2], [1, 1, 2], [3, 1, 3], [2, 3, 1], [3, 1, 3]],
    [[3, 3, 1], [1, 3, 2], [2, 3, 2], [1, 3, 3], [1, 3, 2]], [[2, 1, 3], [3, 2, 1], [3, 1, 1], [2, 2, 3], [1, 1, 1]],
    [[2, 2, 3], [2, 2, 3], [1, 3, 3], [1, 3, 1], [2, 1, 1]], [[2, 3, 2], [1, 2, 1], [2, 3, 1], [3, 1, 3], [3, 1, 3]],
    [[3, 1, 1], [3, 3, 2], [1, 3, 1], [3, 1, 2], [2, 1, 2]], [[3, 2, 1], [2, 3, 2], [2, 1, 2], [2, 3, 3], [2, 1, 2]]]
a = [[[-2.7, -2.1, -1.5], [-2.3, -1.8, -1.5], [-2.9, -2.0, -1.5], [-2.5, -2.1, -1.8], [-2.9, -1.9, -1.6]],
    [[-2.9, -2.7, -2.4], [-2.9, -2.7, -2.5], [-3.0, -2.9, -1.7], [-2.8, -1.5, -1.3], [-1.8, -1.7, -1.4]],
    [[-3.0, -2.5, -1.5], [-2.5, -2.3, -1.5], [-2.9, -2.3, -1.7], [-2.1, -1.7, -1.6], [-2.6, -2.2, -1.3]],
    [[-3.0, -1.6, -1.3], [-2.5, -1.6, -1.2], [-1.9, -1.8, -1.5], [-2.6, -2.0, -1.8], [-2.9, -2.6, -2.3]],
    [[-2.4, -1.4, -1.1], [-2.9, -2.2, -2.1], [-2.4, -2.1, -1.4], [-1.8, -1.4, -1.3], [-2.9, -1.5, -1.2]],
```

```
    [[-2.5, -2.4, -1.2], [-1.8, -1.3, -1.1], [-2.2, -1.9, -1.3], [-2.6, -1.4, -1.1], [-3.0, -1.6, -1.3]],
    [[-2.8, -1.2, -1.1], [-2.6, -2.3, -2.1], [-2.0, -1.4, -1.2], [-3.0, -1.8, -1.7], [-2.7, -2.5, -1.9]],
    [[-2.9, -2.6, -1.9], [-2.3, -2.2, -1.6], [-2.5, -1.6, -1.5], [-2.4, -2.0, -1.8], [-2.0, -1.9, -1.4]],
    [[-3.0, -2.0, -1.5], [-3.0, -2.9, -2.0], [-2.4, -2.1, -1.3], [-2.5, -2.2, -1.1], [-2.6, -1.3, -1.1]],
    [[-3.0, -1.5, -1.4], [-2.8, -1.9, -1.3], [-2.5, -2.1, -1.8], [-2.7, -1.4, -1.2], [-2.8, -1.7, -1.3]],
    [[-2.9, -2.8, -2.4], [-2.9, -2.0, -1.4], [-2.5, -1.2, -1.1], [-2.5, -1.7, -1.6], [-3.0, -2.9, -2.4]],
    [[-3.0, -1.9, -1.7], [-2.5, -2.3, -1.5], [-2.4, -2.3, -1.9], [-1.9, -1.7, -1.4], [-2.7, -1.7, -1.3]],
    [[-1.8, -1.6, -1.1], [-2.2, -2.2, -1.1], [-2.7, -2.1, -1.7], [-3.0, -2.8, -2.5], [-2.9, -2.2, -1.7]],
    [[-2.9, -2.5, -1.3], [-2.9, -2.6, -1.1], [-2.2, -2.1, -1.7], [-2.8, -1.8, -1.7], [-2.7, -2.5, -2.3]],
    [[-3.0, -2.2, -1.9], [-2.7, -1.8, -1.3], [-3.0, -2.5, -1.4], [-3.0, -2.3, -1.1], [-1.7, -1.3, -1.2]],
    [[-1.9, -1.7, -1.6], [-2.8, -2.2, -2.0], [-2.9, -1.4, -1.1], [-2.8, -1.6, -1.4], [-2.9, -1.6, -1.6]],
    [[-2.4, -1.6, -1.2], [-2.6, -2.1, -2.1], [-2.0, -1.9, -1.5], [-2.9, -2.4, -1.6], [-2.3, -2.1, -1.5]],
    [[-1.8, -1.4, -1.3], [-3.0, -1.8, -1.1], [-2.2, -1.3, -1.2], [-2.8, -2.2, -2.0], [-2.7, -2.2, -2.1]],
    [[-2.6, -2.2, -2.0], [-2.2, -2.1, -1.4], [-2.1, -1.8, -1.5], [-2.7, -2.5, -2.4], [-2.5, -2.0, -1.7]],
    [[-1.8, -1.5, -1.2], [-2.4, -2.4, -1.5], [-3.0, -2.6, -2.4], [-3.0, -1.6, -1.5], [-2.3, -1.7, -1.1]]]

kindir = 4
kp = [3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0]
kn = [0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4]
kc = [0.0, 0.2, 0.0, 0.0, 0.0]
Td = [180, 220, 300, 320, 380, 410, 450, 500, 550, 590, 640, 690, 720, 760, 830, 870, 930, 970, 1000, 1030]
Tso = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
Tfo = [L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
      L1, L1, L1, L1, L1, L1, L1, L1]
Tsb = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
Tfb = [L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
      L1, L1, L1, L1, L1, L1, L1, L1]
Ctol = [[0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0, 0.0]]
Cwol = [[L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1], [L1, L1, L1, L1, L1],
       [L1, L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1], [L1, L1, L1, 0, L1],
       [L1, L1, L1, 0, L1]]
Cwou = [[L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1], [L1, L1, L1, L3, L1],
       [L1, L1, L1, L3, L1]]
Ctbl = [[0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0]]
Cwbl = [[L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
```

```
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1]]
Ctbu = [[L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [0, 0, 0, 0], [L1, L1, L1, L1]]
Cwbu = [[L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L2, L2, L2, L2], [L1, L1, L1, L1]]


engine = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl,
      Ctbu, Cwbu,
      pr=1, save=1, ret=1, ro=0.01)

#per = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20] #0
#per = [1, 19, 11, 10, 12, 9, 8, 18, 20, 13, 15, 4, 5, 17, 14, 16, 6, 7, 2, 3] #1
#per = [1, 13, 9, 8, 19, 17, 11, 10, 12, 18, 14, 4, 5, 20, 16, 15, 2, 3, 6, 7] #2
#per = [1, 19, 11, 12, 10, 17, 15, 6, 4, 5, 7, 20, 9, 8, 14, 13, 16, 18, 2, 3] #3
#per = [1, 18, 8, 9, 13, 15, 19, 17, 11, 10, 12, 6, 2, 3, 14, 20, 16, 7, 4, 5] #4
per = [1, 2, 3, 6, 10, 11, 12, 15, 14, 7, 16, 18, 17, 20, 13, 4, 5, 9, 8, 19] #5


vobj, NWRr, NWZr, NPRr, NPZr = engine.solve(per)

start_date = datetime.date(2023, 1, 1)

plotGantt(genNazwa(per), start_date, NWRr, NWZr, NPRr, NPZr, brigades=[])
exportToExcel(genNazwa(per), f"{name}_wynik", start_date, NWRr, NWZr, NPRr, NPZr)

C = 0.1
bmn = 8
mt = 3
sbm = 2
sUCBm = 3

et = 3*3600
con = ['1=1', '2>>3', '4>>5', '6>7', '8<>9', "10_11_12"]
MCTSP1 = MCTS(calc_engine=engine, c=C, constrains=con, brutemaxnumber=bmn, exectime=et, mintime=mt,
        genVisualization=0,
        selectbestnodemode=sbm, selectUCBmode=sUCBm, log=1, gtt=1)
```

Załącznik 15: Kod programu w języku Python realizujący przykład 2

```
from PS import PS
from prj import *

'''Stałe'''
L1 = 1e4
L2 = 1e6
L3 = 1e8
L4 = 1e10
L5 = 1e12
L6 = 1e14

'''Zadanie praktyczne 2'''
name = "P2"

kgr = [[35.3, 65.0, 66.8, 60.2], [34.1, 79.7, 50.4, 67.9], [38.5, 77.0, 68.1, 66.7], [32.9, 79.5, 57.8, 68.7],
    [37.9, 62.2, 56.3, 61.7], [35.0, 65.1, 69.7, 66.6], [30.4, 61.1, 53.2, 62.3], [34.2, 66.9, 65.3, 68.2],
    [38.5, 60.8, 65.0, 64.8], [33.4, 60.5, 62.6, 60.9], [33.7, 67.2, 53.6, 66.1], [32.6, 76.5, 69.7, 65.7],
    [37.1, 70.5, 55.4, 63.7], [32.6, 77.1, 60.2, 66.7], [30.1, 75.3, 68.1, 62.1], [30.6, 61.1, 64.7, 69.7],
    [35.3, 62.5, 52.6, 63.4], [36.1, 76.7, 65.4, 63.2], [36.1, 70.1, 61.4, 62.3], [38.0, 62.4, 62.8, 69.1],
    [39.6, 64.0, 50.7, 69.7], [33.1, 65.2, 68.2, 64.4], [33.5, 76.7, 63.3, 68.0], [32.6, 71.6, 66.8, 62.6],
    [33.2, 64.2, 58.5, 69.7], [34.1, 76.9, 50.7, 60.7], [38.3, 73.4, 55.6, 60.6]]
tgr = [[17.0, 49.0, 31.0, 43.0], [15.0, 45.0, 37.0, 33.0], [16.0, 50.0, 37.0, 41.0], [11.0, 49.0, 37.0, 31.0],
    [14.0, 49.0, 37.0, 32.0], [19.0, 47.0, 31.0, 32.0], [17.0, 49.0, 32.0, 34.0], [12.0, 42.0, 35.0, 40.0],
    [20.0, 43.0, 32.0, 31.0], [17.0, 44.0, 37.0, 34.0], [19.0, 48.0, 33.0, 31.0], [13.0, 47.0, 36.0, 43.0],
    [19.0, 46.0, 38.0, 46.0], [17.0, 45.0, 34.0, 46.0], [17.0, 42.0, 35.0, 35.0], [12.0, 42.0, 38.0, 43.0],
    [13.0, 40.0, 35.0, 46.0], [14.0, 44.0, 38.0, 45.0], [15.0, 50.0, 30.0, 45.0], [20.0, 46.0, 33.0, 48.0],
    [16.0, 48.0, 36.0, 43.0], [16.0, 45.0, 37.0, 31.0], [15.0, 46.0, 34.0, 43.0], [18.0, 46.0, 39.0, 45.0],
    [18.0, 48.0, 30.0, 43.0], [11.0, 42.0, 34.0, 30.0], [16.0, 48.0, 31.0, 30.0]]
t = [[[3, 1, 1], [2, 2, 1], [1, 1, 3], [2, 2, 1]], [[3, 2, 2], [3, 2, 1], [3, 2, 1], [2, 1, 1]],
    [[3, 3, 1], [2, 3, 1], [2, 1, 1], [3, 1, 2]], [[1, 2, 3], [2, 1, 3], [3, 1, 1], [1, 3, 3]],
    [[3, 1, 2], [1, 1, 3], [3, 1, 3], [3, 2, 2]], [[2, 2, 3], [2, 2, 2], [2, 1, 2], [1, 1, 3]],
    [[3, 3, 2], [1, 3, 3], [2, 2, 2], [3, 3, 1]], [[1, 3, 3], [1, 2, 1], [2, 2, 2], [3, 3, 3]],
    [[1, 3, 2], [3, 1, 2], [1, 3, 3], [2, 3, 1]], [[2, 1, 1], [2, 2, 1], [3, 2, 2], [1, 3, 2]],
    [[1, 3, 2], [2, 3, 2], [1, 3, 3], [3, 3, 1]], [[1, 2, 1], [1, 1, 2], [3, 1, 1], [1, 2, 2]],
    [[1, 3, 3], [3, 3, 1], [1, 1, 3], [3, 3, 2]], [[1, 3, 3], [3, 2, 3], [3, 1, 3], [1, 2, 3]],
    [[2, 1, 3], [2, 1, 1], [2, 3, 3], [2, 2, 2]], [[2, 3, 2], [2, 3, 1], [1, 3, 1], [2, 3, 2]],
    [[2, 3, 2], [2, 2, 3], [3, 1, 2], [3, 3, 3]], [[2, 3, 3], [3, 1, 2], [2, 1, 2], [1, 1, 1]],
    [[3, 1, 3], [2, 3, 1], [2, 3, 3], [2, 2, 2]], [[3, 2, 3], [2, 2, 3], [1, 2, 1], [2, 3, 2]],
    [[3, 1, 2], [1, 1, 3], [2, 1, 2], [1, 3, 1]], [[3, 1, 1], [1, 1, 3], [3, 3, 2], [1, 1, 2]],
    [[3, 2, 3], [3, 3, 1], [1, 3, 3], [3, 3, 1]], [[2, 1, 3], [3, 2, 3], [1, 1, 3], [1, 2, 1]],
    [[2, 2, 2], [3, 2, 2], [3, 3, 2], [1, 1, 3]], [[1, 3, 2], [1, 1, 1], [3, 3, 1], [1, 3, 1]],
    [[1, 1, 2], [3, 2, 3], [3, 2, 1], [1, 3, 3]]]
a = [[[-2.8, -2.7, -1.4], [-3.0, -2.6, -1.9], [-2.6, -2.4, -2.2], [-2.4, -2.3, -1.7]],
    [[-2.6, -2.5, -1.8], [-2.7, -1.6, -1.1], [-2.8, -1.6, -1.1], [-2.6, -2.1, -1.5]],
    [[-2.7, -2.2, -1.8], [-2.8, -2.5, -2.3], [-2.6, -2.0, -1.1], [-2.8, -2.7, -1.1]],
    [[-2.7, -1.6, -1.3], [-2.8, -2.7, -2.6], [-2.0, -1.8, -1.2], [-2.9, -2.2, -1.4]],
    [[-2.0, -1.9, -1.5], [-2.2, -2.0, -1.4], [-2.5, -1.5, -1.3], [-2.1, -1.9, -1.6]],
    [[-2.3, -1.9, -1.2], [-2.5, -2.1, -1.5], [-2.7, -2.4, -2.0], [-3.0, -2.1, -1.5]],
    [[-3.0, -2.2, -1.3], [-2.6, -1.3, -1.1], [-2.5, -2.3, -1.7], [-2.4, -1.3, -1.1]],
    [[-3.0, -1.2, -1.1], [-2.5, -2.0, -1.3], [-3.0, -2.7, -2.1], [-2.0, -1.5, -1.1]],
    [[-2.9, -2.6, -2.3], [-2.2, -1.3, -1.2], [-2.9, -2.2, -1.3], [-2.1, -1.6, -1.5]],
    [[-2.6, -2.4, -1.7], [-2.8, -1.7, -1.5], [-2.7, -2.4, -2.2], [-3.0, -2.9, -1.1]],
    [[-2.5, -2.2, -1.2], [-2.0, -1.4, -1.3], [-2.9, -2.4, -1.8], [-2.5, -2.0, -1.6]],
    [[-2.6, -2.1, -1.6], [-1.8, -1.7, -1.3], [-3.0, -1.6, -1.1], [-2.5, -2.4, -1.9]],
    [[-3.0, -2.8, -1.7], [-2.9, -2.1, -1.1], [-2.7, -2.3, -1.5], [-3.0, -2.0, -1.8]],
    [[-2.1, -1.9, -1.1], [-2.8, -2.2, -2.1], [-3.0, -2.0, -1.1], [-3.0, -1.8, -1.3]],
    [[-2.0, -1.6, -1.1], [-1.9, -1.6, -1.5], [-3.0, -2.4, -2.0], [-2.9, -2.1, -1.2]],
    [[-2.9, -2.8, -1.8], [-2.2, -1.5, -1.3], [-2.9, -2.2, -1.4], [-2.5, -2.3, -1.3]],
    [[-3.0, -2.9, -2.7], [-2.8, -2.5, -2.4], [-2.8, -2.0, -1.5], [-2.8, -2.4, -2.1]],
    [[-2.9, -1.7, -1.3], [-2.0, -1.5, -1.3], [-2.3, -2.3, -1.4], [-2.9, -2.3, -1.6]],
    [[-2.5, -1.7, -1.3], [-3.0, -1.4, -1.1], [-2.7, -2.3, -2.3], [-2.3, -1.6, -1.3]],
    [[-1.7, -1.3, -1.1], [-3.0, -2.5, -2.1], [-2.9, -2.7, -2.5], [-1.6, -1.5, -1.2]],
    [[-2.9, -2.8, -1.2], [-2.6, -1.9, -1.8], [-2.1, -1.6, -1.5], [-2.5, -2.1, -1.5]],
    [[-1.5, -1.3, -1.2], [-2.9, -2.6, -1.1], [-2.9, -2.5, -2.4], [-2.0, -1.7, -1.3]],
    [[-3.0, -2.7, -1.5], [-2.4, -2.4, -1.6], [-1.9, -1.3, -1.1], [-3.0, -2.3, -2.1]],
    [[-2.1, -1.9, -1.3], [-2.7, -1.8, -1.2], [-2.0, -1.9, -1.3], [-2.9, -2.8, -2.1]],
    [[-2.1, -1.7, -1.5], [-3.0, -2.8, -2.3], [-2.7, -2.5, -1.6], [-2.6, -2.2, -1.7]],
    [[-2.0, -1.7, -1.1], [-3.0, -1.9, -1.1], [-2.6, -2.4, -2.3], [-2.4, -2.3, -1.6]],
    [[-2.2, -2.0, -1.7], [-2.6, -2.3, -1.6], [-2.8, -1.4, -1.1], [-2.9, -2.2, -2.0]]]
```

```
kindir = 4
kp = [3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0,
    3.0, 3.0, 3.0, 3.0]
kn = [0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4,
    0.4, 0.4, 0.4, 0.4]
kc = [0.0, 0.0, 0.0, 0.0]
Td = [210, 260, 300, 350, 390, 430, 470, 530, 580, 620, 660, 700, 740, 780, 820, 860, 900, 950, 1000, 1040, 1090, 1120,
    1200, 1240, 1260, 1320, 1340]
Tso = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
     0.0, 0.0, 0.0, 0.0, 0.0]
Tfo = [L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
     L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
     L1, L1, L1]
Tsb = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
     0.0, 0.0, 0.0, 0.0, 0.0]
Tfb = [L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
     L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
     L1, L1, L1]
Ctol = [[0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0, 0.0]]
Cwol = [[L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1]]
Ctou = [[L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1]]
Cwou = [[L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1],
      [L1, L1, L1, L1], [L1, L1, L1, L1]]
Ctbl = [[0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
      [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0]]
Cwbl = [[L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
```

```
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1]]
Ctbu = [[L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1]]
Cwbu = [[L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1]]
Ca = []

engine = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl,
        Ctbu, Cwbu,
        pr=1, save=1, ret=1)

#per=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27]#0
#per=[12,11,10,15,13,14,16,18,17,21,19,20,22,24,23,2,3,1,5,6,4,9,8,7,25,26,27]#1
#per=[12,10,11,14,13,15,16,18,17,3,1,2,5,4,6,8,7,9,20,21,19,24,23,22,25,26,27]#2
#per=[10,12,11,1,3,2,13,15,14,16,18,17,5,6,4,8,7,9,20,19,21,24,23,22,25,26,27]#3
per=[12,11,10,14,13,15,17,18,16,2,3,1,4,5,6,9,8,7,20,21,19,23,24,22,25,26,27]#4
#per=[12,10,11,15,13,14,16,17,18,3,2,1,21,19,20,22,23,24,5,6,4,9,8,7,25,26,27]#5


#uszeregowania z przykladu 3
# per = [12, 11, 10, 1, 3, 2, 20, 19, 21, 6, 4, 5, 8, 7, 9, 23, 24, 22, 14, 13, 15, 17, 16, 18, 26, 25, 27] #1
# per = [1, 2, 3, 19, 21, 20, 6, 4, 5, 8, 7, 9, 11, 12, 10, 13, 15, 14, 23, 22, 24, 16, 17, 18, 26, 25, 27] #2
# per = [19, 21, 20, 2, 1, 3, 6, 5, 4, 8, 7, 9, 11, 12, 10, 14, 13, 15, 17, 16, 18, 22, 23, 24, 26, 25, 27] #3
# per = [2, 3, 1, 5, 4, 6, 20, 19, 21, 9, 7, 8, 10, 12, 11, 23, 24, 22, 13, 14, 15, 17, 16, 18, 26, 25, 27] #4
# per = [21, 19, 20, 1, 3, 2, 5, 6, 4, 9, 7, 8, 12, 11, 10, 15, 14, 13, 23, 24, 22, 17, 16, 18, 26, 25, 27]  #5


vobj, NWRr, NWZr, NPRr, NPZr, costs = engine.solve(per)

print(vobj)

print(f"bla: {NPZr[:, 3]}")

start_date = datetime.date(2023, 1, 1)

plotGantt(genNazwa(per), start_date, NWRr, NWZr, NPRr, NPZr, brigades=[])
exportToExcel(genNazwa(per), f"{name}_wynik", start_date, NWRr, NWZr, NPRr, NPZr)

C = 0.1
bmn = 8
mt = 3
sbm = 2
sUCBm = 3

et = 3*3600

con = ['1_2_3',
     '4_5_6',
     '7_8_9',
     '10_11_12',
     '13_14_15',
     '16_17_18',
```

```
    '19_20_21',
    '22_23_24',
    '25_26_27',
    '1>4',
    '4>7',
    '10>13',
    '13>16',
    '19>22',
    '22>25', ]

MCTSP1 = MCTS(calc_engine=engine, c=C, constrains=con, brutemaxnumber=bmn, exectime=et, mintime=mt,
        genVisualization=0,
        selectbestnodemode=sbm, selectUCBmode=sUCBm, log=1, gtt=1)
```

Załącznik 16: Kod programu w języku Python realizujący przykład 3

```
from PS import PS
from MCTS import *
from prj import *
import threading

'''Stałe'''
L1 = 1e4
L2 = 1e6
L3 = 1e8
L4 = 1e10
L5 = 1e12
L6 = 1e14

'''Zadanie praktyczne 3'''
name = "P3"

kgr = [[35.3, 65.0, 66.8, 60.2], [34.1, 79.7, 50.4, 67.9], [38.5, 77.0, 68.1, 66.7], [32.9, 79.5, 57.8, 68.7],
    [37.9, 62.2, 56.3, 61.7], [35.0, 65.1, 69.7, 66.6], [30.4, 61.1, 53.2, 62.3], [34.2, 66.9, 65.3, 68.2],
    [38.5, 60.8, 65.0, 64.8], [33.4, 60.5, 62.6, 60.9], [33.7, 67.2, 53.6, 66.1], [32.6, 76.5, 69.7, 65.7],
    [37.1, 70.5, 55.4, 63.7], [32.6, 77.1, 60.2, 66.7], [30.1, 75.3, 68.1, 62.1], [30.6, 61.1, 64.7, 69.7],
    [35.3, 62.5, 52.6, 63.4], [36.1, 76.7, 65.4, 63.2], [36.1, 70.1, 61.4, 62.3], [38.0, 62.4, 62.8, 69.1],
    [39.6, 64.0, 50.7, 69.7], [33.1, 65.2, 68.2, 64.4], [33.5, 76.7, 63.3, 68.0], [32.6, 71.6, 66.8, 62.6],
    [33.2, 64.2, 58.5, 69.7], [34.1, 76.9, 50.7, 60.7], [38.3, 73.4, 55.6, 60.6]]
tgr = [[17.0, 49.0, 31.0, 43.0], [15.0, 45.0, 37.0, 33.0], [16.0, 50.0, 37.0, 41.0], [11.0, 49.0, 37.0, 31.0],
    [14.0, 49.0, 37.0, 32.0], [19.0, 47.0, 31.0, 32.0], [17.0, 49.0, 32.0, 34.0], [12.0, 42.0, 35.0, 40.0],
    [20.0, 43.0, 32.0, 31.0], [17.0, 44.0, 37.0, 34.0], [19.0, 48.0, 33.0, 31.0], [13.0, 47.0, 36.0, 43.0],
    [19.0, 46.0, 38.0, 46.0], [17.0, 45.0, 34.0, 46.0], [17.0, 42.0, 35.0, 35.0], [12.0, 42.0, 38.0, 43.0],
    [13.0, 40.0, 35.0, 46.0], [14.0, 44.0, 38.0, 45.0], [15.0, 50.0, 30.0, 45.0], [20.0, 46.0, 33.0, 48.0],
    [16.0, 48.0, 36.0, 43.0], [16.0, 45.0, 37.0, 31.0], [15.0, 46.0, 34.0, 43.0], [18.0, 46.0, 39.0, 45.0],
    [18.0, 48.0, 30.0, 43.0], [11.0, 42.0, 34.0, 30.0], [16.0, 48.0, 31.0, 30.0]]
t = [[[3, 1, 1], [2, 2, 1], [1, 1, 3], [2, 2, 1]], [[3, 2, 2], [3, 2, 1], [3, 2, 1], [2, 1, 1]],
    [[3, 3, 1], [2, 3, 1], [2, 1, 1], [3, 1, 2]], [[1, 2, 3], [2, 1, 3], [3, 1, 1], [1, 3, 3]],
    [[3, 1, 2], [1, 1, 3], [3, 1, 3], [3, 2, 2]], [[2, 2, 3], [2, 2, 2], [2, 1, 2], [1, 1, 3]],
    [[3, 3, 2], [1, 3, 3], [2, 2, 2], [3, 3, 1]], [[1, 3, 3], [1, 2, 1], [2, 2, 2], [3, 3, 3]],
    [[1, 3, 2], [3, 1, 2], [1, 3, 3], [2, 3, 1]], [[2, 1, 1], [2, 2, 1], [3, 2, 2], [1, 3, 2]],
    [[1, 3, 2], [2, 3, 2], [1, 3, 3], [3, 3, 1]], [[1, 2, 1], [1, 1, 2], [3, 1, 1], [1, 2, 2]],
    [[1, 3, 3], [3, 3, 1], [1, 1, 3], [3, 3, 2]], [[1, 3, 3], [3, 2, 3], [3, 1, 3], [1, 2, 3]],
    [[2, 1, 3], [2, 1, 1], [2, 3, 3], [2, 2, 2]], [[2, 3, 2], [2, 3, 1], [1, 3, 1], [2, 3, 2]],
    [[2, 3, 2], [2, 2, 3], [3, 1, 2], [3, 3, 3]], [[2, 3, 3], [3, 1, 2], [2, 1, 2], [1, 1, 1]],
    [[3, 1, 3], [2, 3, 1], [2, 3, 3], [2, 2, 2]], [[3, 2, 3], [2, 2, 3], [1, 2, 1], [2, 3, 2]],
    [[3, 1, 2], [1, 1, 3], [2, 1, 2], [1, 3, 1]], [[3, 1, 1], [1, 1, 3], [3, 3, 2], [1, 1, 2]],
    [[3, 2, 3], [3, 3, 1], [1, 3, 3], [3, 3, 1]], [[2, 1, 3], [3, 2, 3], [1, 1, 3], [1, 2, 1]],
    [[2, 2, 2], [3, 2, 2], [3, 3, 2], [1, 1, 3]], [[1, 3, 2], [1, 1, 1], [3, 3, 1], [1, 3, 1]],
    [[1, 1, 2], [3, 2, 3], [3, 2, 1], [1, 3, 3]]]
a = [[[-2.8, -2.7, -1.4], [-3.0, -2.6, -1.9], [-2.6, -2.4, -2.2], [-2.4, -2.3, -1.7]],
    [[-2.6, -2.5, -1.8], [-2.7, -1.6, -1.1], [-2.8, -1.6, -1.1], [-2.6, -2.1, -1.5]],
    [[-2.7, -2.2, -1.8], [-2.8, -2.5, -2.3], [-2.6, -2.0, -1.1], [-2.8, -2.7, -1.1]],
    [[-2.7, -1.6, -1.3], [-2.8, -2.7, -2.6], [-2.0, -1.8, -1.2], [-2.9, -2.2, -1.4]],
    [[-2.0, -1.9, -1.5], [-2.2, -2.0, -1.4], [-2.5, -1.5, -1.3], [-2.1, -1.9, -1.6]],
    [[-2.3, -1.9, -1.2], [-2.5, -2.1, -1.5], [-2.7, -2.4, -2.0], [-3.0, -2.1, -1.5]],
    [[-3.0, -2.2, -1.3], [-2.6, -1.3, -1.1], [-2.5, -2.3, -1.7], [-2.4, -1.3, -1.1]],
    [[-3.0, -1.2, -1.1], [-2.5, -2.0, -1.3], [-3.0, -2.7, -2.1], [-2.0, -1.5, -1.1]],
    [[-2.9, -2.6, -2.3], [-2.2, -1.3, -1.2], [-2.9, -2.2, -1.3], [-2.1, -1.6, -1.5]],
    [[-2.6, -2.4, -1.7], [-2.8, -1.7, -1.5], [-2.7, -2.4, -2.2], [-3.0, -2.9, -1.1]],
    [[-2.5, -2.2, -1.2], [-2.0, -1.4, -1.3], [-2.9, -2.4, -1.8], [-2.5, -2.0, -1.6]],
    [[-2.6, -2.1, -1.6], [-1.8, -1.7, -1.3], [-3.0, -1.6, -1.1], [-2.5, -2.4, -1.9]],
    [[-3.0, -2.8, -1.7], [-2.9, -2.1, -1.1], [-2.7, -2.3, -1.5], [-3.0, -2.0, -1.8]],
    [[-2.1, -1.9, -1.1], [-2.8, -2.2, -2.1], [-3.0, -2.0, -1.1], [-3.0, -1.8, -1.3]],
    [[-2.0, -1.6, -1.1], [-1.9, -1.6, -1.5], [-3.0, -2.4, -2.0], [-2.9, -2.1, -1.2]],
    [[-2.9, -2.8, -1.8], [-2.2, -1.5, -1.3], [-2.9, -2.2, -1.4], [-2.5, -2.3, -1.3]],
    [[-3.0, -2.9, -2.7], [-2.8, -2.5, -2.4], [-2.8, -2.0, -1.5], [-2.8, -2.4, -2.1]],
    [[-2.9, -1.7, -1.3], [-2.0, -1.5, -1.3], [-2.3, -2.3, -1.4], [-2.9, -2.3, -1.6]],
    [[-2.5, -1.7, -1.3], [-3.0, -1.4, -1.1], [-2.7, -2.3, -2.3], [-2.3, -1.6, -1.3]],
    [[-1.7, -1.3, -1.1], [-3.0, -2.5, -2.1], [-2.9, -2.7, -2.5], [-1.6, -1.5, -1.2]],
    [[-2.9, -2.8, -1.2], [-2.6, -1.9, -1.8], [-2.1, -1.6, -1.5], [-2.5, -2.1, -1.5]],
    [[-1.5, -1.3, -1.2], [-2.9, -2.6, -1.1], [-2.9, -2.5, -2.4], [-2.0, -1.7, -1.3]],
    [[-3.0, -2.7, -1.5], [-2.4, -2.4, -1.6], [-1.9, -1.3, -1.1], [-3.0, -2.3, -2.1]],
    [[-2.1, -1.9, -1.3], [-2.7, -1.8, -1.2], [-2.0, -1.9, -1.3], [-2.9, -2.8, -2.1]],
```

```
    [[-2.1, -1.7, -1.5], [-3.0, -2.8, -2.3], [-2.7, -2.5, -1.6], [-2.6, -2.2, -1.7]],
    [[-2.0, -1.7, -1.1], [-3.0, -1.9, -1.1], [-2.6, -2.4, -2.3], [-2.4, -2.3, -1.6]],
    [[-2.2, -2.0, -1.7], [-2.6, -2.3, -1.6], [-2.8, -1.4, -1.1], [-2.9, -2.2, -2.0]]]
kindir = 4
kp = [3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0, 3.0,
     3.0, 3.0, 3.0, 3.0]
kn = [0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4, 0.4,
     0.4, 0.4, 0.4, 0.4]
kc = [0.0, 0.0, 0.0, 0.0]
Td = [210, 260, 300, 350, 390, 430, 470, 530, 580, 620, 660, 700, 740, 780, 820, 860, 900, 950, 1000, 1040, 1090, 1120,
     1200, 1240, 1260, 1320, 1340]
Tso = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
      0.0, 0.0, 0.0, 0.0, 0.0]
Tfo = [L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
      L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
      L1, L1, L1]
Tsb = [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
      0.0, 0.0, 0.0, 0.0, 0.0]
Tfb = [L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
      L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1, L1,
      L1, L1, L1]
Ctol = [[0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0], [0.0, 0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0, 0.0]]
Cwol = [[L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1],
       [L1, L1, L1, L1], [L1, L1, L1, L1]]
Ctou = [[L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1],
       [L1, 0, 0, L1], [L1, 0, 0, L1]]
Cwou = [[L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1],
       [L1, L2, L2, L1], [L1, L2, L2, L1]]
Ctbl = [[0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0],
       [0.0, 0.0, 0.0], [0.0, 0.0, 0.0], [0.0, 0.0, 0.0]]
```

```
Cwbl = [[L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1]]
Ctbu = [[L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1]]
Cwbu = [[L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1],
      [L1, L1, L1], [L1, L1, L1], [L1, L1, L1]]
Ca = []

engine = PS(name, tgr, kgr, t, a, kindir, kp, kn, kc, Td, Tso, Tfo, Tsb, Tfb, Ctol, Cwol, Ctou, Cwou, Ctbl, Cwbl,
        Ctbu, Cwbu,
        pr=1, save=1, ret=1)

per = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27] #0
per = [12, 11, 10, 1, 3, 2, 20, 19, 21, 6, 4, 5, 8, 7, 9, 23, 24, 22, 14, 13, 15, 17, 16, 18, 26, 25, 27] #1
per = [1, 2, 3, 19, 21, 20, 6, 4, 5, 8, 7, 9, 11, 12, 10, 13, 15, 14, 23, 22, 24, 16, 17, 18, 26, 25, 27] #2
per = [19, 21, 20, 2, 1, 3, 6, 5, 4, 8, 7, 9, 11, 12, 10, 14, 13, 15, 17, 16, 18, 22, 23, 24, 26, 25, 27] #3
per = [2, 3, 1, 5, 4, 6, 20, 19, 21, 9, 7, 8, 10, 12, 11, 23, 24, 22, 13, 14, 15, 17, 16, 18, 26, 25, 27] #4
#per = [21, 19, 20, 1, 3, 2, 5, 6, 4, 9, 7, 8, 12, 11, 10, 15, 14, 13, 23, 24, 22, 17, 16, 18, 26, 25, 27] #5


#uszeregowania z przykladu 2
per=[12,11,10,15,13,14,16,18,17,21,19,20,22,24,23,2,3,1,5,6,4,9,8,7,25,26,27]#1
per=[12,10,11,14,13,15,16,18,17,3,1,2,5,4,6,8,7,9,20,21,19,24,23,22,25,26,27]#2
per=[10,12,11,1,3,2,13,15,14,16,18,17,5,6,4,8,7,9,20,19,21,24,23,22,25,26,27]#3
per=[12,11,10,14,13,15,17,18,16,2,3,1,4,5,6,9,8,7,20,21,19,23,24,22,25,26,27]#4
per=[12,10,11,15,13,14,16,17,18,3,2,1,21,19,20,22,23,24,5,6,4,9,8,7,25,26,27]#5


vobj, NWRr, NWZr, NPRr, NPZr = engine.solve(per)

print(f"bla: {NPZr[:, 3]}")

start_date = datetime.date(2023, 1, 1)

plotGantt(genNazwa(per), start_date, NWRr, NWZr, NPRr, NPZr, brigades=[])
exportToExcel(genNazwa(per), f"{name}_wynik", start_date, NWRr, NWZr, NPRr, NPZr)

C = 0.1
bmn = 8
mt = 3
sbm = 2
sUCBm = 3

et = 3*3600

con = ['1_2_3',
     '4_5_6',
     '7_8_9',
     '10_11_12',
     '13_14_15',
```

```
    '16_17_18',
    '19_20_21',
    '22_23_24',
    '25_26_27',
    '1>4',
    '4>7',
    '10>13',
    '13>16',
    '19>22',
    '22>25', ]

MCTSP1 = MCTS(calc_engine=engine, c=C, constrains=con, brutemaxnumber=bmn, exectime=et, mintime=mt,
        genVisualization=0,
        selectbestnodemode=sbm, selectUCBmode=sUCBm, log=1, gtt=1)
```

8. Spis tabel, rysunków i załączników

8.1. Spis rysunków

Rysunek 1: Schemat rozprawy 13
Rysunek 2: Wyniki ATAL S.A. w latach 2015-2021 21
Rysunek 3: Wizualizacja osiedla Skwer Harmonia 21
Rysunek 4: Wyniki Dom Development S.A. w latach 2015-2021. 23
Rysunek 5: Plan inwestycji Osiedle Wilno. 23
Rysunek 6: Wyniki Echo Investment S.A. w latach 2015-2021. 24
Rysunek 7: Liczba mieszkań oddanych do użytkowania w latach 2015-2021. 25
Rysunek 8: Schemat modelu interaktywnego planowania pracy w systemie pracy potokowej 30
Rysunek 9: Liczba publikacji w zależności od wyszukiwanych słów kluczowych w przeglądarce Scopus w obszarze badań inżynierskich 45
Rysunek 10: Podział przedsięwzięć budowlanych ze względu na metody organizacji pracy. 48
Rysunek 11: Schematy metod realizacji przedsięwzięć budowlanych. 49
Rysunek 12: Poglądowe przedstawienie idei potokowych metod organizacji 50
Rysunek 13: Schemat procesu w MSP 55
Rysunek 14: Elastyczne sprzężenia czasowe. 56
Rysunek 15: Sposób naliczania kar dla sprzężeń. 56
Rysunek 16: Przykładowe dodatkowe elastyczne sprzężenie czasowe 59
Rysunek 17: Schemat metody priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych 63
Rysunek 18: Zależność czasowo-kosztowa procesu wraz z graficzną interpretacją parametrów 73
Rysunek 19. Drzewo opisujące możliwe uszeregowania trzech obiektów 76
Rysunek 20. Procedura metody MCTS 77
Rysunek 21: Przykładowa sieć CPM-COST przedstawiona w postaci MSP 87
Rysunek 22: Rozwiązanie optymalne dla terminu dyrektywnego 35 87
Rysunek 23: Zależność minimalnego kosztu bezpośredniego procesu od terminu dyrektywnego 88
Rysunek 24: Zależność minimalnego kosztu całkowitego od czasu trwania przedsięwzięcia 88

Rysunek 25: Zależność średniej wartości funkcji celu, najlepszej osiągniętej wartości funkcji celu oraz średniej liczby symulacji w zależności od przyjętej wartości parametru C dla wzoru $V3$ i sposobu wyboru węzła $S2$ 100
Rysunek 26: Zależność średniej wartości funkcji celu, najlepszej osiągniętej wartości funkcji celu oraz średniej liczby symulacji w zależności od przyjętej wartości parametru C dla wzoru $V3$ i sposobu wyboru węzła $S2$ 101
Rysunek 27: Prezentacja możliwości modułu użytkowego – eksport danych do tabeli Excel. Terminy uzyskane dla zadania C1 i terminu rozpoczęcia 1.01.2023 r. 104
Rysunek 28 Prezentacja możliwości modułu użytkowego – wykres Gantta. Terminy uzyskane dla zadania C1 i terminu rozpoczęcia 1.01.2023 r. 104
Rysunek 29: Fragment wykresu Gantta dla przykładu 1 (iteracja 5) pomiędzy datami 12.05.2023 a 23.01.2024. 108
Rysunek 30: Harmonogram pochodny dla brygady B4 dla przykładu 1 (iteracja 5). 109
Rysunek 31: Schematyczny widok boczny dla przykładu 2..... 111
Rysunek 32: Widok ogólny wykresu Gantta dla przykładu 2 (iteracja 4)..... 113

8.2.Spis tabel

Tabela 1: Nomenklatura w harmonogramowaniu przedsięwzięć wieloobiektowych 27
Tabela 2: Wybrane publikacje uwzględniające analizę czasową w porządku chronologicznym 34
Tabela 3: Wybrane publikacje uwzględniające analizę kosztową w porządku chronologicznym 40
Tabela 4 Czasy wykonania poszczególnych procesów przez brygady - przykład ilustrujący metodę harmonogramowania priorytetowego 53
Tabela 5 Ciągłość pracy brygad - przykład 53
Tabela 6 Ciągłość pracy na obiektach - przykład 53
Tabela 7 Ciągłość pracy dla brygady 2, brygady 3 i na obiekcie 3 - przykład 54
Tabela 8 Rozwiązanie z uwzględnieniem priorytetów - przykład 55
Tabela 9: Przykład, który prezentuje właściwości sprzężeń czasowych elastycznych 59
Tabela 10: Opracowane zestawy wag parametrów realizujące różne efekty decyzyjne, uwzględniające elastyczne sprzężenia czasowe 81
Tabela 11: Opracowane zestawy wag parametrów realizujące różne efekty decyzyjne, uwzględniające elastyczne dodatkowe sprzężenia czasowe 83
Tabela 12: Opracowane zestawy wag parametrów realizujące różne efekty decyzyjne, uwzględniające elastyczne dodatkowe sprzężenia czasowe 85
Tabela 13: Czasy trwania procesów wykonywanych przez brygady Bj na obiektach Oi... 89
Tabela 14: Otrzymane wyniki dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od A.1 do A.8 89
Tabela 15: Otrzymane wyniki dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od B.1 do B.5 93
Tabela 16: Otrzymane wyniki dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od C.1 do C.3 95
Tabela 17 Analiza eksperymentalna 102
Tabela 18. Porównanie metod MC, SA i MCTS 102
Tabela 19: Zakresy danych do wygenerowania przykładu obliczeniowego 106
Tabela 20: Wyniki uzyskane dla pięciu iteracji (oraz uszeregowania początkowego – iteracja 0) wykonane dla przykładu 1. 106
Tabela 21: Przykładowy przebieg algorytmu dla przykładu 1 (iteracja 5). 107
Tabela 22: Dane wyjściowe modułu użytkowego dla przykładu 1 (iteracja 5). 110

Tabela 23: Wyniki uzyskane dla pięciu iteracji (oraz uszeregowania początkowego) wykonane dla przykładu 2. 112
Tabela 24: Wyniki uzyskane dla pięciu iteracji (oraz uszeregowania początkowego) wykonane dla przykładu 3. 114
Tabela 25: Porównanie przykładu 2 i przykładu 3 115
Tabela 26: Analiza wrażliwości dla przykładu 1. 118
Tabela 27: Analiza wrażliwości dla przykładu 2. 118
Tabela 28: Analiza wrażliwości dla przykładu 3. 119
Tabela 29: Średnia zmiana wartości czasu i kosztu przy zmianie parametrów dla przykładów 1,2,3. 119
Tabela 30: Odchylenie standardowe zmiany wartości czasu i kosztu przy zmianie parametrów dla przykładów 1,2,3. 119

8.3.Spis załączników

Załącznik 1: Kod funkcji realizującą czasowo-kosztowy model priorytetowego harmonogramowania 133
Załącznik 2: Kod dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od A.1-A.8 138
Załącznik 3: Kod dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od A.1-A.8 144
Załącznik 4: Kod dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od B.1-B.5 150
Załącznik 5: Kod realizujący optymalizację dyskretną zmodyfikowaną metodą MCTS .. 154
Załącznik 6: Kod dla przykładu obliczeniowego modelu harmonogramowania priorytetowego dla modeli od C.1-C.3 160
Załącznik 7: Wartość średniej wartości funkcji celu w zależności od parametru C dla wszystkich kombinacji Vi oraz Si 162
Załącznik 8: Wartość średniej wartości funkcji celu w zależności od parametru C dla wszystkich kombinacji Vi oraz Si 163
Załącznik 9: Przebieg 20 różnych iteracji algorytmu MCTS dla $(S2, V3)$ oraz wartości parametru $C = 0{,}1$. 164
Załącznik 10: Przykładowy przebieg algorytmu MCTS dla zadania ta092 169
Załącznik 11: Kod programu realizujący moduł użytkowy – eksport danych do Excel oraz rysowanie harmonogramu Gantta. 170
Załącznik 12: Przykładowy przebieg algorytmu MCTS dla zadania ta041 z uwzględnieniem ograniczeń kolejności wykonania obiektów 172
Załącznik 13: Kod programu w języku Python realizujący moduł użytkowy 173
Załącznik 14: Kod programu w języku Python realizujący przykład 1 175
Załącznik 15: Kod programu w języku Python realizujący przykład 2 178
Załącznik 16: Kod programu w języku Python realizujący przykład 3 182

9. Streszczenie

W pracy zaprezentowano opracowaną metodę priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych. Na podstawie wywiadów z osobami odpowiedzialnymi za planowanie i harmonogramowanie przedsięwzięć wieloobiektowych, przeglądu literatury oraz analizy rynku przedsięwzięć wieloobiektowych wykazano, że: brakuje świadomości wśród inżynierów budownictwa w zakresie wpływu kolejności wykonania obiektów na czas i koszt przedsięwzięcia wieloobiektowego, występują braki w oprogramowaniu pomocnym przy harmonogramowaniu przedsięwzięć wieloobiektowych, brakuje modeli uwzględniających wszystkie rodzaje kosztów, ważnych z punktu widzenia realizacji przedsięwzięcia wieloobiektowego oraz brakuje modeli, które w sposób elastyczny pozwolą definiować ograniczenia technologiczno-organizacyjne. W związku ze zidentyfikowanymi brakami została opracowana metoda priorytetowego harmonogramowania wieloobiektowych przedsięwzięć budowlanych, która pozwala uwzględnić ograniczenia technologiczno-organizacyjne w sposób elastyczny, na podstawie listy priorytetów. W funkcji celu uwzględniono koszty bezpośrednie, koszty pośrednie, koszty niedotrzymania terminów dyrektywnych, nagrody za wcześniejsze zrealizowanie obiektów, koszty nieciągłości pracy brygad. Skuteczność działania opracowanej metody została potwierdzona wielopoziomową weryfikacją i walidacją. W ramach walidacji operacyjnej przeprowadzono analizę wrażliwości, która wykazała wrażliwość modelu na zmianę parametrów „czas graniczny” oraz „termin dyrektywny”. Opracowana metoda udowodniła swoją skuteczność minimalizując koszty całkowite przedsięwzięcia przy dochowaniu ograniczeń technologiczno-organizacyjnych zgodnie z założonymi efektami planistycznymi, co zostało zaprezentowane na przykładach obliczeniowych. Wykorzystując opracowaną metodę udało się zredukować koszty całkowite przedsięwzięcia średnio o ok. 3,6%, a czas o średnio ok. 0,7% przy dochowywaniu ograniczeń technologiczno-organizacyjnych w sposób priorytetowy. W pracy przedstawiono modele matematyczne oraz kody programu napisanego w języku Python. Opracowana metoda może być pomocna dla inżynierów i kierowników budowy do harmonogramowania przedsięwzięć wieloobiektowych przy uwzględnieniu ograniczeń technologiczno-organizacyjnych, ograniczeń kolejności wykonania obiektów oraz minimalizującą koszty całkowite przedsięwzięcia. W celu poprawy działania metody, zaproponowano następujące kierunki dalszych badań: uwzględnienie danych w postaci probabilistycznej lub rozmytej, rozszerzenie metody o możliwość wykonywania konkretnych procesów przez więcej niż jedną wyspecjalizowaną brygadę roboczą, opracowanie interfejsu granicznego użytkownika.

10. Abstract

This dissertation presents an original repetitive construction project priority scheduling method. Based on interviews with persons in charge of planning and scheduling repetitive projects, and a literature review and an analysis of the market for such projects, it was demonstrated that: there is a lack of awareness among civil engineers of the impact of execution sequences on the time and cost of a repetitive project, there is a shortage of software that assists in repetitive project scheduling, there is a lack of models that consider all types of costs important for the execution of a repetitive project, and there is a lack of models that flexibly define technological and organisational constraints. In view of the deficiencies identified, a method for repetitive construction project priority scheduling was developed, which allows technological and organisational constraints to be considered in a flexible manner, based on a priority list. Included in the objective function are direct costs, indirect costs, costs of failing to meet contractual deadlines, rewards for the early completion of project units, and the cost of discontinuous crew work. The effectiveness of the method was confirmed using multi-level verification and validation. As part of the operational validation, a sensitivity analysis was performed, which showed the sensitivity of the model to a change in 'cut-off time' and ‘contractual deadline’ parameters. The method developed proved its effectiveness in minimising total project cost while adhering to technological and organisational constraints as per the assumed planning outcomes, which was demonstrated using calculation examples. Using the method proposed, it was possible to reduce total project cost by an average of about 3.6% and total project duration by an average of about 0.7%, while adhering to technological and organisational constraints as a matter of priority. The dissertation presents mathematical models and program code written in Python. The method proposed can be helpful to site engineers and directors for scheduling repetitive projects, taking into account technological and organisational constraints, sequencing constraints and minimising total project cost. To improve the method’s performance, the following future research trajectories were proposed: inclusion of data in probabilistic or fuzzy form, extension of the method to allow specific processes to be performed by more than one specialised crew, development of a graphical user interface.