AKADEMIA GÓRNICZO-HUTNICZA

im. Stanisława Staszica w Krakowie

WYDZIAŁ ZARZĄDZANIA

STUDIA STACJONARNE

KIERUNEK: Zarządzanie i Inżynieria Produkcji

PRACA DYPLOMOWA

Magisterska

Maciej Gacek

Porównanie wybranych reguł priorytetu harmonogramowania produkcji

A comparison of selected production scheduling priority rules

Promotor: dr inż. Antoni Korcyl

Zatwierdzam do rejestracji i dopuszczam do obrony

.....

Data i podpis promotora

Kraków, 2020

Spis treści

Wstęp

Harmonogramowanie jest jednym z ważniejszych zadań zarządzającego, ponieważ prowadząc działalność menadżerską niejednokrotnie dochodzi do konieczności podjęcia decyzji co do wykonania poszczególnych działań w jakimś określonym terminie. Harmonogramowanie produkcji zdefiniować można jako problem wyznaczenia w czasie i przestrzeni takiego przedziału dostępnych zasobów produkcyjnych, który będzie w stanie zapewnić wykonanie tych zadań produkcyjnych przy najlepszym możliwym wykorzystaniu tych zasobów.

Praca porusza tematykę harmonogramowania ze względu na jego istotę w procesie produkcyjnym.

Celem poniższej pracy było porównanie wybranych reguł priorytetu harmonogramowania produkcji. W tym celu oprócz wstępu teoretycznego na temat harmonogramowania oraz reguł priorytetu zaimplementowano w języku Python 3.9 algorytmy harmonogramujące produkcję przy pomocy wybranych reguł priorytetu. Do porównania posłużyły czasy trwania algorytmów otrzymywanych przy użyciu wcześniej wymienionych reguł, średnie czasy zakończenia obróbki poszczególnych detali oraz opóźnienia zakończenia obróbki detali względem dyrektywnego terminu zakończenia obróbki.

Plan poniższej pracy jest następujący. W rozdziale 1. przedstawiono podstawy harmonogramowania produkcji oraz metody harmonogramowania. Dodatkowo opisano w jaki sposób opracowuje się harmonogramy pracy maszyn.

W rozdziale 2. przedstawiono pięć wybranych prostych reguł priorytetu harmonogramowania produkcji, ilustrując je rozwiązaniem przykładowego zadania.

W rozdziale 3. nastąpiło porównanie reguł priorytetu opisanych w rozdziale 2. Przedstawiono zestawy założeń, dla których działał program. Wykonano eksperymenty obliczeniowe, a następnie porównano otrzymane przez nie wyniki. Pod koniec tego rozdziału pojawiły się wnioski wynikające z wcześniej wykonanej pracy.

W rozdziale 4. W postaci załączników znajduje się kod programu użytego w poniższej pracy dyplomowej.

1. Harmonogramowanie produkcji

Ogólnie, harmonogramowanie produkcji można zdefiniować jako problem wyznaczenia takiego rozdziału w czasie i przestrzeni dostępnych zasobów produkcyjnych, który zapewni wykonanie zadań produkcyjnych przy najlepszym wykorzystaniu tych zasobów [10].

Harmonogramowanie jest jednym z ważniejszych zadań zarządzającego, ponieważ prowadząc działalność menadżerską niejednokrotnie dochodzi do konieczności podjęcia decyzji co do wykonania poszczególnych działań w jakimś określonym terminie. Harmonogramy stosować można do „rozmieszczenia w czasie" wielu różnorodnych operacji, czynności czy też zadań, a horyzont czasowy utworzonego harmonogramu nie powinien się różnić od horyzontu planistycznego. W procesie zarządzania produkcja bardzo szczególna rolę odgrywają krótkoterminowe plany operacyjne oraz powiązane z nimi harmonogramy operacyjne. Harmonogramy operacyjne różnią się od siebie w zależności czy mamy do czynienia z produkcją na magazyn (MTS, ang. make to stock), czy produkcją na zamówienie (MTO, ang. make to order). W sytuacji gdy produkcja odbywa się na magazyn najczęściej przygotowywany harmonogram jest stały w danym, określonym horyzoncie planowania. Zależność ta podyktowana jest faktem, że praca takiego systemu produkcyjnego oraz jego urządzeń skupiona jest tylko na konkretnym produkcie (oczywiście w danym horyzoncie planistycznym). W takim przypadku głównym problemem jest balansowanie przepływu materiału (równoważenie) poprzez indywidualne stanowiska robocze bądź ich grupy. Umożliwia to zadowalające obciążenie stanowisk, a harmonogram ulega zmianie dopiero w kolejnym okresie planowania, w którym już najczęściej produkuje się inny produkt. Harmonogramy tego rodzaju nazywa się harmonogramami statycznymi i powinny one zwłaszcza skupiać się na czasie odbioru oraz przekazaniu na magazyn określonej partii produkcyjnej. Sytuacja zaczyna być bardziej skomplikowana kiedy chcemy produkować jednocześnie na magazyn kilka różnych produktów. W takim wypadku metoda opracowywania harmonogramu jest podobna jak w przypadku produkcji MTO [2].

W przypadku wykonywania produkcji na zamówienie (MTO) produkty są zamawiane przez klientów w różnej wielkości zamówienia oraz w różnych odmianach, które dodatkowo napływają w różnych terminach. Taki brak stabilizacji (występujący

w zdecydowaniem większym stopniu niż kiedy towary produkowane są na magazyn) sprawia, że harmonogram musi być wielokrotnie korygowany, głównie z konieczności zmiany obciążeń stanowisk roboczych. Wynikać to może z napływu nowego zlecenia o wyższym priorytecie, awarii jednej z maszyn czy chociażby choroby jednego z pracowników. Harmonogramy opracowywane w takiej sytuacji nazywane są harmonogramami dynamicznymi, gdyż zakłada się jego korekty w zależności od otrzymywanych informacji z procesu wytwarzania oraz działu przygotowania produkcji. MTO oznacza występowanie więcej niż jednej operacji oczekujących na użycie określonego wyposażenia. W takiej sytuacji należy znaleźć najlepszą kombinację zleceń produkcyjnych oraz zasobów [2].

Harmonogramy tworzone są za pomocą dwóch podstawowych metod:

a) Harmonogramowanie w przód, w przypadku w którym znana jest data rozpoczęcia pierwszej operacji oraz znane są czasy trwania operacji, możliwe jest wyznaczenie daty zakończenia poszczególnych operacji, a co za tym idzie terminu wykonania zlecenia.
b) Harmonogramowanie wstecz, w przypadku w którym znany jest wymagany termin zakończenia zlecenia oraz poszczególne czasy operacji, możliwe jest wyznaczenie najpóźniejszego rozpoczęcia pierwszej operacji.

Wydawać się może, że problem harmonogramowania jest bardzo prosty oraz łatwy do rozwiązania. Sytuacja komplikuje się jednak kiedy mamy do czynienia z większą liczbą zleceń do zrealizowania, nie ma możliwości zmiany kolejności wykonywaniach operacji czy obciążenia poszczególnych stanowisk determinują możliwość wykonania danej operacji. W rzeczywistości występuje wiele tego typu zakłóceń oraz „zmiennych", a co za tym idzie opracowanie „optymalnego" harmonogramu jest zagadnieniem skomplikowanym i niezwykle złożonym [2, 5].

1.1. Opracowanie harmonogramów pracy maszyn

Mając do czynienia z licznym zbiorem czynności produkcyjnych oraz operacji ich realizacja powinna być wzajemnie umiejscowiona w czasie (pod względem kolejności), z punktu widzenia robotników, wykorzystania czasu pracy maszyny czy rytmiczności spływu detali. W zależności od wariantu szeregowania tworzą się różne reperkusje organizacyjne oraz ekonomiczne, zauważalne przede wszystkim poprzez wydłużanie się lub skracanie cyklu produkcyjnego. Dlatego projektując harmonogramy należy uwzględnić poniższe zagadnienia

a) W harmonogramie tak należy ustalić kolejność detalooperacji, aby łączny czas przezbrojeń oraz cykl produkcyjny osiągały najkorzystniejsze wielkości.
b) Należy usytuować operacje detalów względem siebie oraz względem pozostałych detalooperacji tak, aby cykle produkcyjne poszczególnych detali były odpowiednie.

Każda z decyzji o wyborze jednego z wariantów, który ma być obowiązujący podczas tworzenia harmonogramu , jest zasadniczo decyzją o nadaniu danej operacji lub danemu wyrobowi priorytetu. W literaturze wymienia się nawet ponad sto różnych reguł nadawania priorytetów, które wykorzystują różne kryteria optymalizacji czy różne parametry przebiegu procesu produkcyjnego w komórce produkcyjnej [4].

Niezależnie od tego jaką regułę priorytetu wybraliśmy do ustalenia wykonywania określonych detalooperacji należy pamiętać, że tworzenie harmonogramu produkcji nawet dla małej komórki produkcyjnej jest bardzo złożonym problemem. Ze względu na ogromną liczbę wariantów, nawet przy pomocy programowania matematycznego nie zawsze możliwe jest uzyskanie rozwiązania optymalnego „metodą prób i błędów". W związku z tym najczęściej stosowane są praktyczne usprawnienia, przy których uzyskiwane są minimalne długości cyklów produkcyjnych detali w danej komórce produkcyjnej.

Podczas projektowania harmonogramu pamiętać należy, że:

a) Zasadniczy wpływ na długość cyklu produkcyjnego wyrobów oraz łączną ich długość ma kolejność ułożenia wyrobów w harmonogramie.

b) W miarę układania wyrobu w harmonogramie wzrasta wskaźnik wydłużenia cyklu. Najmniejszym wskaźnikiem wydłużenia cyklu produkcyjnego odznacza się wyrób umieszczany, jako pierwszy w harmonogramie.
c) Wzrost obciążenia stanowisk roboczych wpływa na wzrost średniego wskaźnika wydłużania cyklu.
d) Podczas tworzenia harmonogramu niejednokrotnie musimy wybierać czy lepiej wydłużyć cykl produkcyjny, czy podzielić partię produkcyjną.
e) Likwidacja mikro przerw (zapewnienie ciągłości) wpływa na wydłużenie cyklu produkcyjnego [3].

1.2. Metody harmonogramowania produkcji

Odpowiedzią na problemy harmonogramowania produkcji są różne metody szeregowania zadań produkcyjnych. Metody te dzieli się na odpowiednie grupy, a podział ten podyktowany jest różnorodnością algorytmów, które stosuje się do tworzenia poszczególnych metod [6].

Pierwszą z tych grup stanowią metody, które odnaleźć mają dokładne rozwiązanie zadania harmonogramowania, zalicza się do nich:

- przeszukiwanie zupełne i losowe,
- programowanie całkowitoliczbowe.

Metoda przeszukiwania zupełnego oparta jest na rachunku kombinatorycznym. Dzięki niej można określić liczbę wszystkich możliwych harmonogramów dopuszczalnych, następnie za pomocą tej metody możliwe jest wybranie z wyznaczonego zbioru wszystkich możliwych rozwiązań. Niestety w złożonych przypadkach metoda ta daje bardzo trudne do wykorzystania wyniki. W związku z tym często stosowana jest metoda przeszukiwania losowego. Zamiast generować wszystkie dopuszczalne rozwiązania to w losowy sposób tworzone są tylko niektóre wyniki, czyli w momencie wystąpienia problemu podejmowana jest losowa decyzja. Natomiast programowanie całkowitoliczbowe odnajduje zastosowanie tylko, gdy mamy do czynienia z zadaniami czysto teoretycznymi. Jednak ze względu na rozmiar

wyników, dużą liczbę zmiennych oraz długi czas obliczeń jest ona rzadko stosowana [6].

Kolejną z kolei grupę stanowi metoda podziałów i ograniczeń, która także znajduje zastosowanie przy rozwiązywaniu problemów harmonogramowania. Metoda ta opiera się na utworzeniu z rozwiązań dopuszczalnych drzewa decyzyjnego. Wierzchołek stanowi harmonogram pewnego podzbioru operacji, z kolei gałęzie tworzone są z możliwych operacji. Istotą tej metody jest pozbywanie się kolejnych gałęzi na podstawie określonej funkcji celu. Wygenerowane najlepsze rozwiązanie jest górną granicą funkcji, a dolna granica jest obliczana na podstawie drogi od miejsca bieżącego, aż do wierzchołka drzewa. Jednak ze względu na długi czas poszukiwania rozwiązań tą metodą jest ona stosowana stosunkowo rzadko [6].

Także systemy ekspertowe odnajdują zastosowanie przy problematyce szeregowania zadań produkcyjnych. Podczas stosowania ich rozpatruje się dwa podejścia:

- algorytmiczne,
- uwzględniające zmianę sformułowania problemu.

Jeżeli chodzi o podejście algorytmiczne to problemy definiowane są przy pomocy standardowego programowania matematycznego. Kiedy problem zostanie sformułowany (niezmienny w trakcie rozwiązywania) uruchamiany jest odpowiedni algorytm, którego zadaniem jest wyszukanie dopuszczalnego wyniku. Z kolei drugie podejście pozwala zmodyfikować problem, a celem algorytmu jest znalezienie nie tyle rozwiązania optymalnego, co satysfakcjonującego [6].

Kolejną grupą metod są metody heurystyczne, u podstaw, których stoją algorytmy heurystyczne. Algorytmy te dzielą się na:

- algorytmy jednokrotne (systemy dyspozytorskie i niedyspozytorskie),
- algorytmy korygujące (symulowane wyżarzanie, wspinanie na szczyt, przeszukiwanie tabu).

Dla algorytmów jednokrotnych nie istnieje możliwość zmiany harmonogramu. Jeżeli konkretna operacja posiada czas rozpoczęcia, to nie istnieje możliwość jego późniejszej zmiany. Natomiast następne operacje są umieszczane w harmonogramie

nie ingerując we wcześniejsze operacje. Algorytmy korygujące z kolei dopuszczają możliwość zmiany czasu operacji włączonej wcześniej do harmonogramu. Zmiana ta polegać może na przyśpieszeniu lub opóźnieniu terminu operacji. Zmiany te mogą również dotyczyć całej grupy operacji. Możliwa jest również wielokrotna korekta terminów rozpoczęcia operacji [6].

Ostatnią grupę stanowią metody zaliczane do algorytmów ewolucyjnych. Zaliczamy do nich:

- algorytmy genetyczne,
- programowanie i strategie ewolucyjne,
- systemy klasyfikatorowe,
- programowanie genetyczne.

Sama idea tych algorytmów powstała w oparciu o już obecne w świecie zjawiska biologiczne. Organizmy żywe, aby przeżyć muszą dostosować się do otaczających je warunków. Algorytmy ewolucyjne to nic innego jak systemy komputerowe, które działają na podobnych zasadach, co organizmy żywe. Algorytmy ewolucyjne tworzą populacje osobników, którymi mogą być wektory liczb rzeczywistych, łańcuchy znaków, macierze, a nawet harmonogramy produkcji. Osobniki te mogą się dalej rozmnażać, krzyżować, mutować oraz przystosowywać do określonych kryteriów. Oczywiście podejście do ewolucji omawianych algorytmów jest wielkim uproszczeniem zjawisk przyrodniczych, to jednak potrafią one zapewnić wysoką jakość procesu poszukiwania rozwiązań dopuszczalnych[6].

2. Wybrane reguły priorytetu

Aby omówić sterowanie czy harmonogramowanie produkcji za pomocą reguł priorytetu należy zdefiniować kilka pojęć:

- zadanie to nic innego jak zlecenie produkcyjne na wykonanie wyrobu złożonego lub prostego,

- kolejka to zbiór pojedynczych operacji, które należą do różnych zadań i oczekują przed stanowiskiem roboczym,

- wskaźnik priorytetu to numeryczne cechy poszczególnych operacji, oczekujących w kolejce przed stanowiskiem na wykonanie. W ogólnym przypadku wskaźnik ten jest funkcją parametrów, które opisują zadania oraz ich stan podczas realizacji procesu produkcyjnego,

- reguła priorytetu jest funkcją, która z kolei przyporządkowuje poszczególnym operacjom oczekującym w kolejce przed konkretnym stanowiskiem wielkość zwaną wskaźnikiem priorytetu oraz wskazuje operację z maksymalną (lub minimalną) wartością wskaźnika definiując tym samym w jakiej kolejności wykonane mają być poszczególne operacje. Definicję reguły priorytetu można zapisać następująco:

$$P_{ij}(t) = \min_{(i,j)\in A(t)} \{z_{ij}(t)\} \tag{2.1}$$

gdzie:

$P_{ij}(t)$ - wskaźnik priorytetu operacji *J* zadania *I* mającej priorytet w chwili *t*,

$z_{ij}(t)$ - wskaźnik priorytetu operacji *j* zadania *i* w chwili *t*,

$A(t)$ - zbiór operacji oczekujących na wykonanie przed danym stanowiskiem w chwili *t*.

- priorytet jest właściwością jednej z operacji oczekujących w kolejce przed stanowiskiem, która została wytypowana spośród nich do wykonania, jako pierwsza w wyniku zastosowania reguły priorytetu [3].

Wskaźniki priorytetu dla poszczególnych operacji w kolejce obliczane są na podstawie danych dotyczących tych operacji oraz zadań, do których te operacje należą. Ewentualnie

wykorzystywane są również informacje o innych operacjach i zadaniach, które znajdują się w kolejkach innych, niż ta która jest aktualnie rozpatrywana. W zależności od zakresu wykorzystywanych informacji można podzielić reguły priorytetu na lokalne i ogólne [3].

Lokalne reguły priorytetu *L* to funkcje tylko tych parametrów operacji i zadań, które oczekują w kolejce przed stanowiskiem, które rozpatrujemy [3].

Ogólne reguły priorytetu *O* to funkcje parametrów operacji i zadań, które oczekują przed innymi stanowiskami, niż to które rozpatrujemy lub operacji i zadań, które akurat są wykonywane na tych stanowiskach, bądź operacji i zadań które oczekują przed rozpatrywanym stanowiskiem. W zależności od możliwości wykonania zmiany wartości wskaźnika priorytetu w czasie reguły priorytetu możemy podzielić na statyczne i dynamiczne [3].

Statyczne reguły priorytetu *S* to reguły, dla których wskaźniki priorytetu są funkcjami parametrów niezależnych od upływającego czasu. Wartości tych wskaźników nie zmieniają się wraz z upływającym czasem, są stałe. Przykładem takiego parametru jest jednostkowy czas operacji [3].

Dynamiczne reguły priorytetu *D* to reguły, dla których wskaźniki priorytetu są funkcjami parametrów zależnych od upływającego czasu. Wartości tych wskaźników zmieniają się wraz z upływającym czasem. Za przykład takich parametrów mogą posłużyć: suma stanowiskochłonności operacji, czas oczekiwania w kolejce, długość kolejki [3].

Jeśli poprzez P, P_1, P_2, P_3 oznaczymy proste (elementarne) parametry, które występują w regułach priorytetu (np. dyrektywny termin zakończenia zadania, czas operacji), poprzez *b* wagę danego parametru, a poprzez *W* określony warunek (np., jeżeli czas oczekiwania operacji w danej kolejce jest dłuższy niż *Q*, to…, jeżeli liczba operacji w danej kolejce jest większa niż *N*, to…), to możemy wyróżnić następujące przykładowe, ogólne postacie funkcji wskaźnika priorytetu:

$$A\colon z_{ij}(t) = P \tag{2.2}$$

$$B\colon z_{ij}(t) = P_1 + P_2 \tag{2.3}$$

$$z_{ij}(t) = bP_1 + (1-b)P_2 \tag{2.4}$$

$$z_{ij}(t) = \sum_k P_k \tag{2.5}$$

$$z_{ij}(t) = \frac{P_1}{P_2} \tag{2.6}$$

$$z_{ij}(t) = \frac{P_1}{P_2^b} \tag{2.7}$$

$$C: z_{ij}(t) = \begin{cases} P_1, jeśl\ warunek\ W\ jest\ spełniony \\ P_2, jeśl\ warunek\ W\ nie\ jest\ spełniony \end{cases} \tag{2.8}$$

Biorąc pod uwagę złożoność funkcji wskaźnika priorytetu dzielimy reguły na:

a) Proste – z funkcjami o postaci *A*, w tych regułach wykorzystywany jest tylko jeden parametr.
b) Złożone – z funkcjami o postaci *B*, funkcje te są sumą, ilorazem bądź iloczynem kilku parametrów (wykorzystuje się również wagi).
c) Kombinowane – z funkcjami o postaci *C,* w funkcjach tych występują również warunki decydujące czy dany człon reguły będzie funkcjonował albo kiedy należy zastosować daną regułę.
d) Heurystyczne – wykorzystują one reguły priorytetu proste lub złożone oraz pewne zasady, które są oparte na doświadczeniu planistycznym. Aby je opisać wykorzystuje się skomplikowane algorytmy, które uwzględniają wiele warunków logicznych. W niektórych sytuacjach takich reguł nie udaje się opisać funkcjami matematycznymi i są one opisane tylko werbalnie [3, 5].

Różne podziały reguł priorytetu przedstawione powyżej obrazują również stopnie trudności związane z ich stosowaniem. Obrazują one pracochłonność, częstotliwość, złożoność obliczeniową oraz charakteryzują jednocześnie zakresy wymaganych informacji. Tak na prawdę nie licząc reguł prostych, pozostałe wymagają wsparcia techniki komputerowej [3, 5].

Oczywiście stosowanie reguł priorytetu jest dokonywane z założeniem uzyskania określonych celów, które zarazem są kryteriami skuteczności ich działania, należą do nich:

- minimalizacja odchyleń faktycznych terminów zakończenia zadań od planowanych (lub dyrektywnych) terminów zakończenia,
- minimalizacja cykli produkcyjnych zadań,
- minimalizacja zakresu prac wykonywanych w komórce produkcyjnej [3].

W literaturze wymienia się nawet ponad 100 reguł priorytetu. W związku z tym istotną kwestią jest dobór odpowiedniej reguły do określonego problemu harmonogramowania produkcji. Dobierając reguły priorytetu bierze się pod uwagę podstawowe trzy czynniki, które wpływają na skuteczność działania oraz możliwość zastosowania tych reguł. Do tych czynników należą:

- sposób napływu zadań do komórki produkcyjnej,
- cel zastosowania reguły priorytetu wynikający z kryterium oceny skuteczności jej działania,
- zakres wymaganych informacji, który wpływa na rodzaj potrzebnych środków organizacyjno-technicznych [3].

Dane wejściowe dla przykładowego zadania, które zostanie rozwiązane przez każdą z później wymienionych prostych reguł priorytetu, w celu ich ilustracji:

Rozważmy zadanie z 8 detalami (A, B, …, H), 5 stanowiskami(1, 2, …, 5) oraz z 5 operacjami technologicznymi (10, 20, …, 50). Każdy detal obrabiany jest dokładnie jeden raz przez każdą z operacji w kolejności rosnącej (od 10 do 50). Czas jednostkowy poszczególnych operacji technologicznych dla każdego detalu oraz wielkość partii produkcyjnej przedstawiono w tabeli 2.1. Kolejność wykonywania
operacji na poszczególnych stanowiskach przedstawia tabela 2.2. Dyrektywny termin zakończenia obróbki każdego z detali przedstawia tabela 2.3. Tabela 2.4 przedstawia zestawienie stanowisk z operacjami poszczególnych detali, jakie będą na nich wykonane oraz dodatkowo czas wykonywania partii detalu przez daną operację. Dyrektywny termin zakończenia obróbki każdego z detali przedstawia tabela 2.4. Kolejność napłynięcia detali do kolejki przedstawia z kolei tabela 2.5.

Tabela 2.1 Czasy jednostkowe operacji, wielkość partii

Detal	Czasy jednostkowe operacji technologicznych					Wielkość partii S
	10	20	30	40	50	
A	0,04	0,08	0,04	0,08	0,06	**50**
B	0,032	0,032	0,04	0,032	0,04	**125**
C	0,03	0,07	0,02	0,02	0,03	**100**
D	0,024	0,04	0,024	0,024	0,04	**125**
E	0,04	0,02	0,04	0,05	0,04	**100**
F	0,06	0,04	0,02	0,05	0,03	**100**
G	0,015	0,025	0,02	0,015	0,01	**200**
H	0,032	0,032	0,048	0,032	0,024	**125**

Źródło: Opracowanie własne

Tabela 2.2 Kolejność wykonywania operacji

Detal	Kolejność wykonywania operacji				
	10	20	30	40	50
A	1	2	3	4	5
B	2	5	1	3	4
C	1	3	5	4	2
D	4	2	5	3	1
E	4	1	3	5	2
F	5	4	1	2	3
G	3	5	4	1	2
H	3	1	2	5	4

Źródło: Opracowanie własne

Tabela 2.3 Zestawienie stanowisk z operacjami, czas obróbki przez operację

Detal	Nr stanowiska/nr operacji				
	1	**2**	**3**	**4**	**5**
A	10	20	30	40	50
	2	4	2	4	3
B	30	10	40	50	20
	5	4	4	5	4
C	10	50	20	40	30
	3	3	7	2	2
D	50	20	40	10	30
	5	5	3	3	3
E	20	50	30	10	40
	2	4	4	4	5
F	30	40	50	20	10
	2	5	3	4	6
G	40	50	10	30	20
	3	2	3	4	5
H	20	30	10	50	40
	4	6	4	3	4

Źródło: Opracowanie własne

Tabela 2.4 Termin oddania detalu

Detal	Termin
A	15
B	25
C	12
D	12
E	20
F	25
G	15
H	20

Źródło: Opracowanie własne

Tabela 2.5 Kolejność napłynięcia detali do kolejki

Detal	Kolejność
A	6
B	7
C	2
D	1
E	8
F	5
G	3
H	4

Źródło: Opracowanie własne

Układając harmonogram dla powyższego zadania będziemy rozpatrywać stanowisko po stanowisku i „ustawiać" na nim poszczególne detale kierując się wybraną metodą priorytetu.

2.1. Reguła priorytetu dla najkrótszego czasu operacji – NCO

Spośród wszystkich operacji, które oczekują na wykonanie w kolejce przed stanowiskiem należy wybrać operację o najkrótszym czasie wykonania operacji

$$P_{ij}(t) = \min_{(i,j)\in A(t)} \{z_{ij}(t) = t_{ij}\} \tag{2.9}$$

gdzie:

$P_{ij}(t)$ - wskaźnik priorytetu operacji *j* zadania *i* mającej priorytet w chwili *t*,

$z_{ij}(t)$ - wskaźnik priorytetu operacji *j* zadania *i* w chwili *t*,

$A(t)$ - zbiór operacji oczekujących na wykonanie przed danym stanowiskiem w chwili *t*

t_{ij} - czas operacji *j* zadania *i* liczony jako iloczyn jednostkowego czasu operacji *tj* i wielkości produkcyjnej *p*.

Reguła NCO przyporządkowuje operacje w kolejce rosnącymi wartościami wskaźników priorytetu, które w przypadku tej reguły są czasami operacji. Z powodu

niezwykle prostej zasady działania jest ona najczęściej stosowaną w praktyce regułą. Ma ona charakter lokalny, statyczny i istnieje wiele jej modyfikacji i odmian, występuje również jako element składowy w wielu regułach kombinowanych i złożonych [7].

Rozwiązanie zadania:

Dla stanowiska nr *1* rozważamy detale *A* oraz *C*, gdyż tylko te dwa detale na tym stanowisku powinny być obrabiane pierwszą operacją nr *10*. Czas wykonywania partii produkcyjnej detalu *A* jest mniejszy niż *C* (2<3) i to ten obiekt będziemy na tym stanowisku obrabiali jako pierwszy. Na stanowisku nr *2* tylko *B* powinno być obrabiane operacją nr *10* więc to on „trafia" do harmonogramu. Na stanowisku nr *3* ponownie musimy rozważyć dwa detale, tym razem *G* o czasie obróbki partii produkcyjnej 3 oraz *H* o czasie obróbki partii produkcyjnej 4. W związku z tym pierwszym detalem obrabianym na *3* stanowisku będzie *G*. Rozpatrując pozostałe dwa stanowiska w identyczny sposób, otrzymujemy zestawienie detali obrabianych jako pierwsze na poszczególnych stanowiskach.

Tabela 2.6 Zestawienie pierwszego etapu tworzenia harmonogramu dla metody NCO

Stanowisko	Detal
1	A
2	B
3	G
4	D
5	F

Źródło: Opracowanie własne

W następnym kroku znowu zaczynając od stanowiska nr *1* sprawdzamy czy dla, któregoś detalu należy wykonać operację nr *10*. W przypadku stanowiska pierwszego jest to tylko detal *C*, a więc on będzie następnym obrabianym na tym stanowisku. Na drugim stanowisku nie ma więcej detali do obróbki operacją nr *10*, natomiast detal *A* oraz *D* czekają na operację nr *20*. Czas wykonywania partii produkcyjnej tego pierwszego jest krótszy, więc to on jako pierwszy trafia do harmonogramu.

Rozpatrując w ten sposób kolejne stanowiska, aż do „wykorzystania” wszystkich operacji na wszystkich detalach otrzymujemy poniższe zestawienie kolejności obróbki danego detalu na konkretnym stanowisku

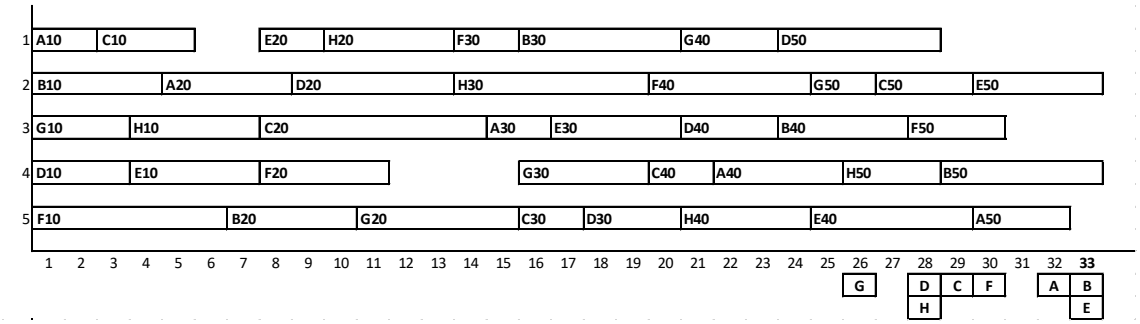
Tabela 2.7 Kolejność obrabianych detali na poszczególnych stanowiskach – reguła NCO

Kolejność operacji na stanowiskach NCO								
1	A	C	E	H	F	B	G	D
2	B	A	D	H	F	G	C	E
3	G	H	C	A	E	D	B	F
4	D	E	F	G	C	A	H	B
5	F	B	G	C	D	H	E	A

Źródło: Opracowanie własne

Otrzymane rezultaty nakładamy na wykres, zakładamy brak przezbrojeń czy przerw, pamiętając o tym, że obróbkę detalu na kolejnym stanowisku rozpocząć możemy dopiero po zakończeniu jego obróbki na stanowisku poprzednim.

Rysunek 2.1 Wykres Gantta dla - reguła NCO



Źródło: Opracowanie własne

Na powyższym wykresie możemy zaobserwować po ilu okresach skończy się obróbka poszczególnych detali oraz kiedy zakończy się cały proces (po 33 okresie).

2.2. Reguła priorytetu dla najdłuższego czasu operacji – NDCO

Spośród wszystkich operacji, które oczekują na wykonanie w kolejce przed stanowiskiem należy wybrać operację o najdłuższym czasie wykonania operacji

$$P_{ij}(t) = \min_{(i,j)\in A(t)} \{z_{ij}(t) = -t_{ij}\} \tag{2.10}$$

Oznaczenia jak dla równania (2.9).

NDCO jest regułą lokalną oraz statyczną. Maksymalizuje ona niemal wszystkie wielkości, które były minimalizowane przez regułę NCO, takie jak średnie odchylenie terminów zakończenia zadania od dyrektywnych terminów ich zakończenia, średni cykl produkcyjny, średni wskaźnik wydłużenia cykli produkcyjnych czy średnia liczbę zadań oczekujących w kolejce. Regułę tę stosować można w przypadku wysokiego stopnia obciążenia komórki produkcyjnej, a zależy nam na szybkim zmniejszeniu tego obciążenia. Reguła ta preferuje bowiem najbardziej stanowiskochłonne operacje [7].

Rozwiązanie zadania:

Dla stanowiska nr *1* rozważamy detale *A* oraz *C*, gdyż tylko te dwa detale na tym stanowisku powinny być obrabiane pierwszą operacją nr *10*. Czas wykonywania partii produkcyjnej detalu *C* jest większy niż *A* (3>2) i to ten obiekt będziemy na tym stanowisku obrabiali jako pierwszy. Na stanowisku nr *2* tylko *B* powinno być obrabiane operacją nr *10* więc to on „trafia" do harmonogramu. Na stanowisku nr *3* ponownie musimy rozważyć dwa detale, tym razem *G* o czasie obróbki partii produkcyjnej 3 oraz *H* o czasie obróbki partii produkcyjnej 4. W związku z tym pierwszym detalem obrabianym na *3* stanowisku będzie *H*. Rozpatrując pozostałe dwa stanowiska w identyczny sposób, otrzymujemy zestawienie detali obrabianych jako pierwsze na poszczególnych stanowiskach.

Tabela 2.8 Zestawienie pierwszego etapu tworzenia harmonogramu – reguła NDCO

Stanowisko	Detal
1	C
2	B
3	H
4	E
5	F

Źródło: Opracowanie własne

W następnym kroku znowu zaczynając od stanowiska nr *1* sprawdzamy czy dla, któregoś detalu należy wykonać operację nr *10*. W przypadku stanowiska pierwszego jest to tylko detal *C*, a więc on będzie następnym obrabianym na tym stanowisku. Na drugim stanowisku nie ma więcej detali do obróbki operacją nr *10*, natomiast detal *A* oraz *D* czekają na operację nr *20*. Czas wykonywania partii produkcyjnej tego drugiego jest dłuższy, więc to on jako pierwszy trafia do harmonogramu. Rozpatrując w ten sposób kolejne stanowiska, aż do „wykorzystania" wszystkich operacji na wszystkich detalach otrzymujemy poniższe zestawienie kolejności obróbki danego detalu na konkretnym stanowisku

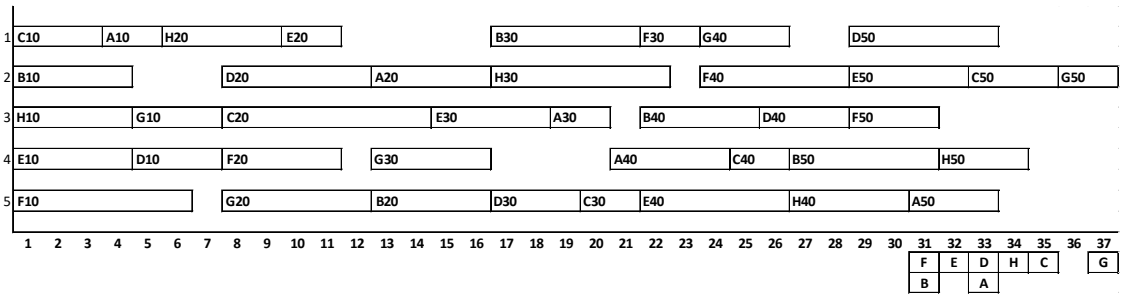
Tabela 2.9 Kolejność obrabianych detali na poszczególnych stanowiskach - reguła NDCO

Kolejność operacji na stanowiskach NDCO								
1	C	A	H	E	B	F	G	D
2	B	D	A	H	F	E	C	G
3	H	G	C	E	A	B	D	F
4	E	D	F	G	A	C	B	H
5	F	G	B	D	C	E	H	A

Źródło: Opracowanie własne

Otrzymane rezultaty nakładamy na wykres, zakładamy brak przezbrojeń czy przerw, pamiętając o tym, że obróbkę detalu na kolejnym stanowisku rozpocząć możemy dopiero po zakończeniu jego obróbki na stanowisku poprzednim.

Rysunek 2.2 Wykres Gantta dla - reguła NDCO



Źródło: Opracowanie własne

Na powyższym wykresie możemy zaobserwować po ilu okresach skończy się obróbka poszczególnych detali oraz kiedy zakończy się cały proces (po 37 okresie).

2.3. Reguła priorytetu dla najwcześniejszego dyrektywnego terminu zakończenia zadania – NTD

Spośród wszystkich operacji, które oczekują na wykonanie w kolejce przed stanowiskiem należy wybrać operację z najwcześniejszym dyrektywnym terminem zakończenia zadania, do którego ta operacja należy

$$P_{ij}(t) = \min_{(i,j)\in A(t)} \{z_{ij}(t) = d_i\} \tag{2.11}$$

gdzie:

d_i – dyrektywny termin zakończenia zadania *i,*

Pozostałe oznaczenia jak dla równania (2.9).

NTD jest regułą lokalną oraz statyczną. Zgodnie z nią operacje porządkowane są według niemalejących dyrektywnych terminów zakończenia zadań, czyli

$$d_1 \leq d_2 \leq \ldots \leq d_n$$

Jest to reguła, która w najprostszy sposób uwzględnia dyrektywne terminy zakończania zadań. Znaczący wpływ na efektywność tej reguły podczas wykonywania

zadań w komórce produkcyjnej ma sposób, w jaki określane są dyrektywne terminy. Najczęściej wynikają one bezpośrednio z cyklu produkcyjnego wyrobu złożonego i określane są przez planistów komórki nadrzędnej. Terminy te powinny być wykonalne i uzasadnione. Zagadnienie to jest bezpośrednio powiązane z prognozowaniem czasu przebywania zadań w komórce produkcyjnej. Gdyby istniała możliwość dokładnego przewidzenia czasów przebywania zadań w komórce to uwzględniając te czasy można by określić dyrektywny termin zakończenia dla każdego zadania. Jednak czas przebywania zadania w komórce zależy od rodzaju i liczby innych zadań wykonywanych w komórce w tym samym czasie, w którym znajduje się tam rozpatrywane zadanie. W związku z tym praktycznie nigdy nie jest możliwe dokładne przewidzenie czasu przebywania zadania w komórce [7].

Rozwiązanie zadania:

Dla stanowiska nr *1* rozważamy detale *A* oraz *C*, gdyż tylko te dwa detale na tym stanowisku powinny być obrabiane pierwszą operacją nr *10*. Dyrektywny termin zakończenia detalu *C* jest krótszy niż detalu *A* (12<15) i to ten obiekt będziemy na tym stanowisku obrabiali jako pierwszy. Na stanowisku nr *2* tylko *B* powinno być obrabiane operacją nr *10* więc to on „trafia" do harmonogramu. Na stanowisku nr *3* ponownie musimy rozważyć dwa detale, tym razem *G* o dyrektywnym terminie zakończenia równym 15 oraz *H* o dyrektywnym terminie zakończenia równym 20. W związku z tym pierwszym detalem obrabianym na *3* stanowisku będzie *G*. Rozpatrując pozostałe dwa stanowiska w identyczny sposób, otrzymujemy zestawienie detali obrabianych jako pierwsze na poszczególnych stanowiskach.

Tabela 2.10 Zestawienie pierwszego etapu tworzenia harmonogramu - reguła NTD

Stanowisko	Detal

1	C
2	B
3	G
4	D
5	F

Źródło: Opracowanie własne

W następnym kroku znowu zaczynając od stanowiska nr *1* sprawdzamy czy dla, któregoś detalu należy wykonać operację nr *10*. W przypadku stanowiska pierwszego jest to tylko detal *C*, a więc on będzie następnym obrabianym na tym stanowisku. Na drugim stanowisku nie ma więcej detali do obróbki operacją nr *10*, natomiast detal *A* oraz *D* czekają na operację nr *20*. Dyrektywny termin zakończenia tego drugiego jest mniejszy, więc to on, jako pierwszy trafia do harmonogramu. Rozpatrując w ten sposób kolejne stanowiska, aż do „wykorzystania" wszystkich operacji na wszystkich detalach otrzymujemy poniższe zestawienie kolejności obróbki danego detalu na konkretnym stanowisku.

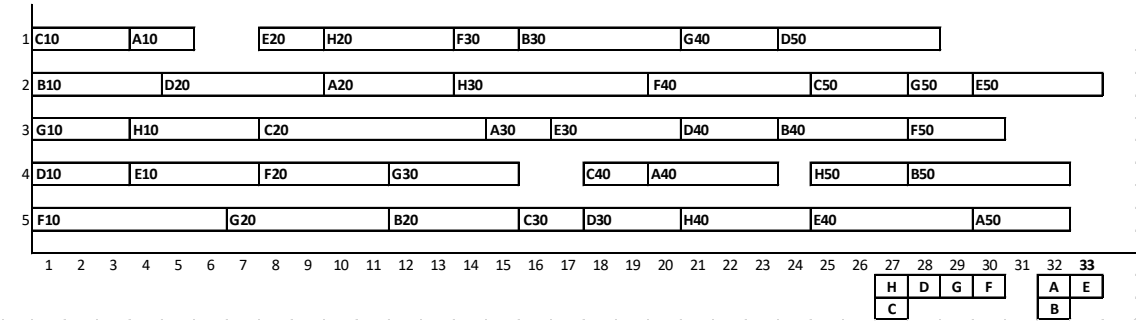
Tabela 2.11 Kolejność obrabianych detali na poszczególnych stanowiskach – reguła NTD

Kolejność operacji na stanowiskach NTD								
1	C	A	E	H	F	B	G	D
2	B	D	A	H	F	C	G	E
3	G	H	C	A	E	D	B	F
4	D	E	F	G	C	A	H	B
5	F	G	B	C	D	H	E	A

Źródło: Opracowanie własne

Otrzymane rezultaty nakładamy na wykres, zakładamy brak przezbrojeń czy przerw, pamiętając o tym, że obróbkę detalu na kolejnym stanowisku rozpocząć możemy dopiero po zakończeniu jego obróbki na stanowisku poprzednim.

Rysunek 2.3 Wykres Gantta - reguła NTD



Źródło: Opracowanie własne

Na powyższym wykresie możemy zaobserwować po ilu okresach skończy się obróbka poszczególnych detali oraz kiedy zakończy się cały proces (po 33 okresie).

2.4. Reguła priorytetu dla operacji pierwszej w kolejce – PPPO

Spośród wszystkich operacji, które oczekują na wykonanie w kolejce przed stanowiskiem należy wybrać operację, która pierwsza przybyła do kolejki

$$P_{ij}(t) = \min_{(i,j)\in A(t)} \{z_{ij}(t) = R_{ij}\} \tag{2.12}$$

gdzie:

R_{ij} – termin przybycia operacji j zadania i do kolejki przed rozpatrywanym stanowiskiem,

Pozostałe oznaczenia jak dla równania (2.9).

PPPO jest regułą lokalną i statyczną. To najprostsza reguła priorytetu rozpatrywana w teorii kolejek. Jest ona nazywana również pierwszy przybył pierwszy obsłużony lub inaczej regulaminem naturalnym. Wyniki otrzymywane dzięki tej metodzie często służą do porównania wyników otrzymanych poprzez użycie innych reguł. Reguła ta często stosowana jest do wyboru operacji, jeżeli przy zastosowaniu innej reguły, więcej niż jedna operacja została zakwalifikowana do wykonania na danym stanowisku [7].

Rozwiązanie zadania:

Dla stanowiska nr *1* rozważamy detale *A* oraz *C*, gdyż tylko te dwa detale na tym stanowisku powinny być obrabiane pierwszą operacją nr *10*. Jako pierwszy do kolejki spłynął detal *C* (2<6) i to ten obiekt będziemy na tym stanowisku obrabiali jako pierwszy. Na stanowisku nr *2* tylko *B* powinno być obrabiane operacją nr *10* więc to on „trafia" do harmonogramu. Na stanowisku nr *3* ponownie musimy rozważyć dwa detale, tym razem *G*, który pojawił się w kolejce jako 3 oraz *H,* który pojawił się w kolejce jako 4. W związku z tym pierwszym detalem obrabianym na *3* stanowisku będzie *G*. Rozpatrując pozostałe dwa stanowiska w identyczny sposób, otrzymujemy zestawienie detali obrabianych jako pierwsze na poszczególnych stanowiskach.

Tabela 2.12 Zestawienie pierwszego etapu tworzenia harmonogramu – reguła PPPO

Stanowisko	Detal
1	C
2	B
3	G
4	D
5	F

Źródło: Opracowanie własne

W następnym kroku znowu zaczynając od stanowiska nr *1* sprawdzamy czy dla, któregoś detalu należy wykonać operację nr *10*. W przypadku stanowiska pierwszego jest to tylko detal *C*, a więc on będzie następnym obrabianym na tym stanowisku. Na drugim stanowisku nie ma więcej detali do obróbki operacją nr *10*, natomiast detal *A* oraz *D* czekają na operację nr *20*. Jako pierwszy do harmonogramu trafia detal *D* gdyż jako pierwszy napłynął do kolejki. Rozpatrując w ten sposób kolejne stanowiska, aż do „wykorzystania" wszystkich operacji na wszystkich detalach otrzymujemy poniższe zestawienie kolejności obróbki danego detalu na konkretnym stanowisku.

Tabela 2.13 Kolejność obrabianych detali na poszczególnych stanowiskach - reguła PPPO

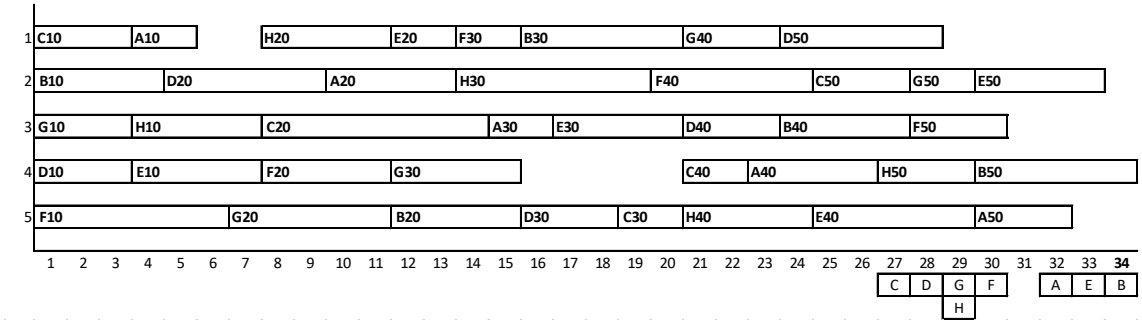
Kolejność operacji na stanowiskach PPPO								
1	C	A	H	E	F	B	G	D
2	B	D	A	H	F	C	G	E

3	G	H	C	A	E	D	B	F
4	D	E	F	G	C	A	H	B
5	F	G	B	D	C	H	E	A

Źródło: Opracowanie własne

Otrzymane rezultaty nakładamy na wykres, zakładamy brak przezbrojeń czy przerw, pamiętając o tym, że obróbkę detalu na kolejnym stanowisku rozpocząć możemy dopiero po zakończeniu jego obróbki na stanowisku poprzednim.

Rysunek 2.4 Wykres Gantta - reguła PPPO



Źródło: Opracowanie własne

Na powyższym wykresie możemy zaobserwować po ilu okresach skończy się obróbka poszczególnych detali oraz kiedy zakończy się cały proces (po 34 okresie).

2.5. Reguła priorytetu dla operacji wybranej losowo - LOS

Spośród wszystkich operacji, które oczekują na wykonanie w kolejce przed stanowiskiem należy wybrać losowo operację

$$P_{ij}(t) = \min_{(i,j)\in A(t)} \{z_{ij}(t) = X_{ij}\} \tag{2.13}$$

gdzie:

X_{ij} – wartość losowa przydzielona do operacji *j* zadania *i,*

Pozostałe oznaczenia jak dla równania (2.9).

LOS jest regułą lokalną i statyczną. Wartości wskaźników mogą być przydzielane z różnych rozkładów zmiennych losowych. Ta reguła często służy jako element odniesienia podczas badania innych reguł oraz do wyboru operacji, jeżeli przy zastosowaniu innej reguły, więcej niż jedna operacja została zakwalifikowana do wykonania na danym stanowisku [7].

Rozwiązanie zadania:

Dla stanowiska nr *1* rozważamy detale *A* oraz *C*, gdyż tylko te dwa detale na tym stanowisku powinny być obrabiane pierwszą operacją nr *10*. Losowo wybrany zostaje detal *C* i to ten obiekt będziemy na tym stanowisku obrabiali jako pierwszy. Na stanowisku nr *2* tylko *B* powinno być obrabiane operacją nr *10* więc to on „trafia" do harmonogramu. Na stanowisku nr *3* ponownie musimy rozważyć dwa detale, tym *G* oraz *H,* losowo wybrany zostaje detal *G*. Rozpatrując pozostałe dwa stanowiska w identyczny sposób, otrzymujemy zestawienie detali obrabianych jako pierwsze na poszczególnych stanowiskach.

Tabela 2.14 Zestawienie pierwszego etapu tworzenia harmonogramu - reguła LOS

Stanowisko	Detal
1	C
2	B
3	G
4	D
5	F

Źródło: Opracowanie własne

W następnym kroku znowu zaczynając od stanowiska nr *1* sprawdzamy czy dla, któregoś detalu należy wykonać operację nr *10*. W przypadku stanowiska pierwszego jest to tylko detal *C*, a więc on będzie następnym obrabianym na tym stanowisku. Na drugim stanowisku nie ma więcej detali do obróbki operacją nr *10*, natomiast detal *A* oraz *D* czekają na operację nr *20*. Losowo jako pierwszy do harmonogramu wybrany zostaje detal *D*. Rozpatrując w ten sposób kolejne stanowiska, aż do „wykorzystania"

wszystkich operacji na wszystkich detalach otrzymujemy poniższe zestawienie kolejności obróbki danego detalu na konkretnym stanowisku.

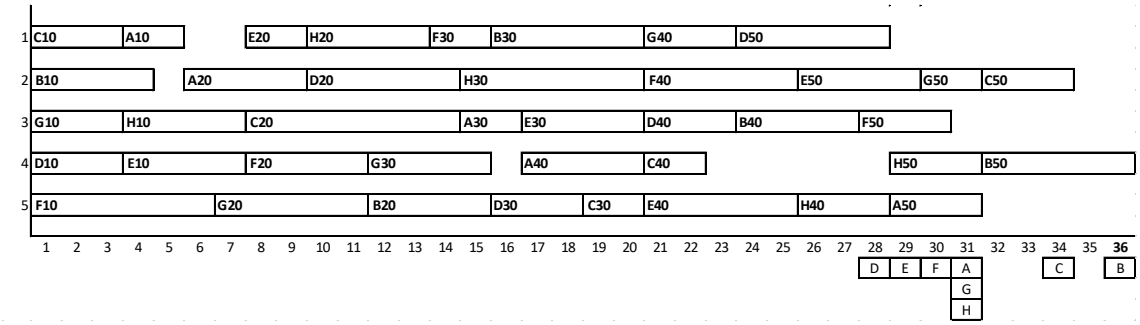
Tabela 2.15 Kolejność obrabianych detali na poszczególnych stanowiskach - reguła LOS

Kolejność operacji na stanowiskach LOS								
1	C	A	E	H	F	B	G	D
2	B	A	D	H	F	E	G	C
3	G	H	C	A	E	D	B	F
4	D	E	F	G	A	C	H	B
5	F	G	B	D	C	E	H	A

Źródło: Opracowanie własne

Otrzymane rezultaty nakładamy na wykres, zakładamy brak przezbrojeń czy przerw, pamiętając o tym, że obróbkę detalu na kolejnym stanowisku rozpocząć możemy dopiero po zakończeniu jego obróbki na stanowisku poprzednim.

Rysunek 2.5 Wykres Gantta - reguła LOS



Źródło: Opracowanie własne

Na powyższym wykresie możemy zaobserwować po ilu okresach skończy się obróbka poszczególnych detali oraz kiedy zakończy się cały proces (po 36 okresie).

3. Porównanie wybranych reguł priorytetu

Poniższy rozdział został poświęcony porównaniu wyników harmonogramowania produkcji przy użyciu reguł priorytetu zaprezentowanych w podrozdziale 2.1. W celu porównania metod stworzyłem autorski algorytm w języku Python 3.8 przy użyciu kompilatora *PyCharm.* Program wykonał obliczenia dla 9 zestawów założeń, a każda reguła, dla każdego zestawu została użyta dla zestawów danych 1000 razy.

3.1. Dane wejściowe do symulacji

Każdy zestaw danych charakteryzuje się określoną liczbą detali oraz liczbą stanowisk, która jest zawsze równa ilości operacji. Każdy detal obrabiany jest dokładnie jeden raz przez każdą z operacji w kolejności rosnącej (10, 20, 30, …, itd.). Czas obróbki detalu dla poszczególnych operacji technologicznych (dla uproszczenia porzucono podział na wielkość partii produkcyjnej oraz czas jednostkowy obróbki danego detalu), dyrektywny termin zakończenia obróbki każdego detalu oraz kolejność wpłynięcia zlecenia na wykonanie danego detalu zostały losowo wygenerowane z rozkładu równomiernego *U*. Stanowiska wykonywania poszczególnych operacji technologicznych poszczególnych detali również zostały wygenerowane losowo z rozkładu równomiernego *U*, tak aby dla każdego detalu występowała każda operacja technologiczna tylko raz. Wszystkie wygenerowane losowo wartości zostały zaokrąglone do liczb całkowitych. W tabeli 3.1 przedstawiono ilość detali, operacji, stanowisk oraz zakresy z jakich losowo generowane były czasy wykonywania poszczególnych detali dla operacji oraz dyrektywne terminy zakończenia obróbki każdego detalu.

Tabela 3.1 Zestawy danych

Nr zestawu	Detale	Operacje	Stanowiska	Czas	Termin
1	8	3	3	2-10	50-70
2	8	3	3	4-20	100-140
3	8	5	5	2-10	50-70
4	8	5	5	4-20	100-140
5	15	5	5	2-10	70-150
6	15	5	5	4-20	140-300
7	15	10	10	2-10	70-150
8	15	10	10	4-20	140-300
9	30	12	12	10-50	300-500

Źródło: Opracowanie własne

3.2. Eksperymenty obliczeniowe

Trzymając się założeń z Tabeli 3.1 wygenerowano losowo po 1000 zestawów danych dla każdego zestawu założeń, a następnie program ułożył harmonogram produkcji kierując się kolejno regułami priorytetu NCO, NDCO, NTD, PPPO oraz LOS. Tabele od 3.2 do 3.10 przedstawiają średnie czasy funkcji C_{max} (czas zakończenia harmonogramu), średni czas zakończenia poszczególnych detali oraz średnie odchylenie od dyrektywnego terminu zakończenia obróbki każdego detalu. Oczywiście podczas harmonogramowania zależy nam na jak najniższej wartości funkcji C_{max} oraz średniego czasu zakończenia poszczególnych elementów. Trzecia wartość przedstawiona w Tabelach 3.2 do 3.10 był liczony ze wzoru:

Termin zakończenia obróbki detalu – dyrektywny termin zakończenia obróbki detalu

W związku z tym zależy nam aby wartość ta była jak najwyższa, gdyż będzie wskazywała w takim przypadku o ile szybciej udało się obrobić dany detal w stosunku do zakładanego terminu. Z kolei im większa wartość na minusie, tym większe opóźnienie w dostarczeniu detalu.

Tabela 3.2 Zestaw 1

Zestaw 1			
	C_{max}	Detale	Termin
NCO	65,53	50,48	9,60
NDCO	66,80	55,74	4,25
NTD	59,37	47,19	12,72
PPPO	60,13	47,65	12,34
LOS	59,94	47,55	12,51

Źródło: Opracowanie własne

Tabela 3.3 Zestaw 2

Zestaw 2			
	C_{max}	Detale	Termin
NCO	132,41	102,13	17,78
NDCO	133,23	110,81	9,12
NTD	118,60	94,59	25,32
PPPO	118,61	94,36	25,63
LOS	118,82	94,45	25,73

Źródło: Opracowanie własne

Tabela 3.4 Zestaw 3

Zestaw 3			
	C_{max}	Detale	Termin
NCO	82,34	68,50	-8,47
NDCO	81,81	71,31	-11,36
NTD	75,19	63,66	-3,68
PPPO	74,42	63,01	-2,88
LOS	74,49	63,20	-3,24

Źródło: Opracowanie własne

Tabela 3.5 Zestaw 4

Zestaw 4			
	C_{max}	Detale	Termin
NCO	163,63	136,69	-16,43
NDCO	163,13	142,23	-22,24
NTD	148,29	125,62	-5,55
PPPO	148,39	126,01	-6,04
LOS	147,86	125,45	-5,37

Źródło: Opracowanie własne

Tabela 3.6 Zestaw 5

Zestaw 5			
	C_{max}	Detale	Termin
NCO	131,50	108,45	1,83
NDCO	131,35	113,55	-3,53
NTD	118,34	99,11	10,95
PPPO	118,26	99,10	11,15
LOS	117,75	98,78	11,22

Źródło: Opracowanie własne

Tabela 3.7 Zestaw 6

Zestaw 6			
	C_{max}	Detale	Termin
NCO	262,43	216,36	3,67
NDCO	262,76	227,33	-7,24
NTD	236,10	197,94	22,11
PPPO	235,81	197,06	23,10
LOS	235,58	197,21	22,36

Źródło: Opracowanie własne

Tabela 3.8 Zestaw 7

Zestaw 7			
	C_{max}	Detale	Termin
NCO	172,33	154,20	-42,63
NDCO	171,60	156,82	-46,05
NTD	157,25	141,74	-30,63
PPPO	156,75	141,62	-30,45
LOS	156,65	141,26	-29,95

Źródło: Opracowanie własne

Tabela 3.9 Zestaw 8

Zestaw 8			
	C_{max}	Detale	Termin
NCO	341,53	306,32	-83,48
NDCO	343,38	314,01	-92,12
NTD	311,80	281,57	-59,34
PPPO	312,35	282,38	-60,50
LOS	311,27	280,87	-58,82

Źródło: Opracowanie własne

Błąd! Nieprawidłowe łącze.Źródło: Opracowanie własne

Już na pierwszy rzut oka widać, że dla większości terminów detale są średnio produkowane z opóźnieniem. Nie ma to jednak żadnego znaczenia z perspektywy porównania wymienionych wcześniej reguł priorytetu. Wynika to tylko i wyłącznie z założeń, z których losowane były dyrektywne terminy zakończenia obróbki poszczególnych detali.

3.3. Porównanie otrzymanych wyników

Porównując wyniki można zauważyć, że metody NCO oraz NDCO odstają od pozostałych trzech reguł. O ile słabe wyniki reguły faworyzującej detale z dłuższymi czasami obróbki nie dziwią, o tyle chociażby w pracy Sobaszka, Goli i Świcia [5] metoda NCO(u nich nazwana STD) wypadła stosunkowo najlepiej, jeżeli za kryterium celu brać

minimalizacje funkcji $C_{max.}$. W Tabeli 3.11 przedstawiono łącznie średnie wyniki z zestawów 1-9.

Tabela 3.10 Średnie wyniki dla Zestawów 1-9

Zestawy 1-9			
	C_{max}	Detale	Termin
NCO	315,34	276,14	-117,78
NDCO	315,71	284,15	-126,07
NTD	284,85	251,50	-93,39
PPPO	284,64	251,38	-93,22
LOS	284,66	251,39	-93,27

Źródło: Opracowanie własne

Widzimy, że metody NTD, PPPO oraz LOS średnio dały niemal takie same wyniki. Różnica między najlepszą, a najgorszą biorąc za kryterium celu kolejno minimalizacje funkcji C_{max}, najszybszy średni czas zakończenia poszczególnych detali oraz najmniejsze spóźnienie wykonania poszczególnych detali w stosunku do wylosowanego dyrektywnego terminu zakończenia obróbki detalu wynosi: 0,21; 0,12 oraz 0,17. Wszystkie te różnice zamykają się więc w poniżej 0,5% średnich otrzymanych wyników. Reguły NTD oraz PPPO od początku do końca harmonogramu faworyzują te same detale, NTD ze względu na termin zakończenia, a PPPO ze względu na kolejność zamówienia – co za tym idzie podobieństwo średnio otrzymanych wyników ma logiczne podstawy.

Nasuwa się więc głównie pytanie, dlaczego średnie wyniki otrzymywane przez regułę NCO wypadają tak słabo? Czy reguła NCO w ogóle dla jakichś zestawów danych dawała najlepsze wyniki? Żeby odpowiedzieć na to pytanie przeanalizowałem dla ilu zestawów danych poszczególne reguły zminimalizowały wartość funkcji C_{max}, a więc ile razy dana reguła ustawiła harmonogram o najkrótszym czasie. Dla tego przykładu użyte zostały założenia z Zestawu numer 3. Wyniki tej analizy przedstawia Tabela 3.12:

Tabela 3.11 Liczba najlepszych rozwiązań

NCO	32
NDCO	44
NTD	176
PPPO	240

LOS	236

Źródło: Opracowanie własne

Powyższa tabela nie uwzględnia zestawów danych, dla których więcej niż jedna reguła dała najlepszy wynik.

Jak się okazuję 32 razy na 728 (ok. 4,5%) reguła NCO zminimalizowała wartość funkcji C_{max}. Jest to jednak wynik najgorszy, nawet reguła NDCO, która według Wróblewskiego, Krawczyńskiego, Kosieradzkiej oraz Kasprzyka [7] powinna maksymalizować wyniki uzyskane przez regułę NCO, wypada w tym zestawieniu lepiej.

Broniąc wyników swojej pracy zdecydowałem się przedstawić jeden z zestawów danych, ukazujący poprawność wyników uzyskiwanych przez stworzony algorytm (program). Opierając się na przytoczonej w pracy teorii najmniej prawdopodobne wydaje się uzyskanie najlepszego wyniku funkcji C_{max} dla reguły NDCO, dlatego postanowiłem przytoczyć jeden z 44 takich przypadków w próbie.

Opis przedstawionych poniżej tabel:

Tabela 3.13 przedstawia wygenerowane losowo numery operacji. Każdy z ośmiu detali (od 0 do 7) ma dla poszczególnych stanowisk (5 – reprezentowanych przez kolumny) przypisane numery operacji (od 10 do 50).

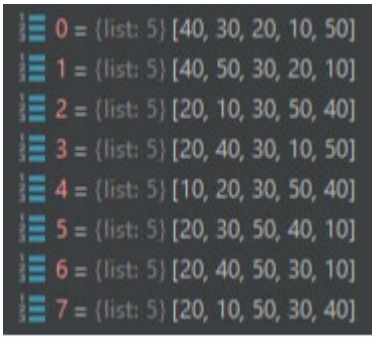
Tabele 3.14 oraz 3.16 przedstawiają kolejność zamówienia detali oraz dyrektywny termin zakończenia obróbki poszczególnych ośmiu detali (od 0 do 7).

Tabelę 3.15 należy czytać dokładnie tak samo jak Tabelę 3.13 z tą różnicą, że wygenerowane w niej wartości losowe reprezentują czas obróbki danego detalu przez daną operację.

Tabele od 3.16 do 3.21 reprezentują harmonogramy otrzymane przy użyciu wcześniej wymienianych reguł priorytetu. Kolumny reprezentują stanowiska (od 0 do 4), a wiersze kolejne czynności wykonywane na poszczególnych stanowiskach. Każde pole wewnątrz tabeli posiada cztery wartości, kolejno: czas rozpoczęcia obróbki, czas zakończenia obróbki, numer detalu, numer operacji jaką obrabiany był detal.

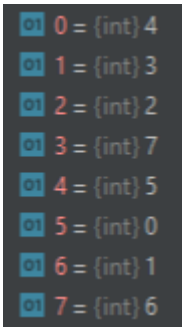
Wygenerowane losowo dane:

Tabela 3.12 Kolejność operacji dla przykładu z podrozdziału 3.4.



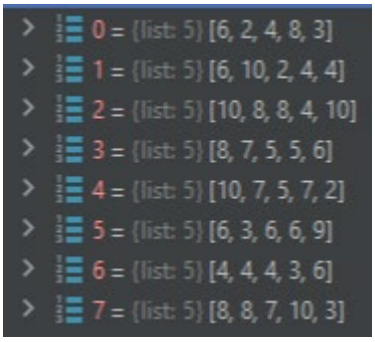
Źródło: Opracowanie własne

Tabela 3.13 Kolejność zamówienia detali dla przykładu z podrozdziału 3.4.



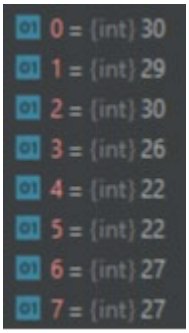
Źródło: Opracowanie własne

Tabela 3.14 Czas obróbki detali dla przykładu z podrozdziału 3.4



Źródło: Opracowanie własne

Tabela 3.15 Dyrektywny termin zakończenia obróbki detalu



Źródło: Opracowanie własne

Otrzymane wyniki:

Tabela 3.16 NCO dla przykładu z podrozdziału 3.4

	0	1	2	3	4
0	(0, 10, 4, 10)	(0, 8, 2, 10)	(13, 17, 0, 20)	(0, 5, 3, 10)	(0, 4, 1, 10)
1	(10, 14, 6, 20)	(8, 16, 7, 10)	(17, 19, 1, 30)	(5, 13, 0, 10)	(4, 10, 6, 10)
2	(19, 25, 5, 20)	(16, 23, 4, 20)	(33, 38, 3, 30)	(13, 17, 1, 20)	(10, 19, 5, 10)
3	(25, 33, 3, 20)	(23, 25, 0, 30)	(38, 43, 4, 30)	(17, 20, 6, 30)	(43, 45, 4, 40)
4	(33, 41, 7, 20)	(25, 28, 5, 30)	(51, 59, 2, 30)	(41, 51, 7, 30)	(51, 54, 7, 40)
5	(41, 51, 2, 20)	(28, 32, 6, 40)	(59, 63, 6, 50)	(51, 57, 5, 40)	(59, 69, 2, 40)
6	(51, 57, 0, 40)	(38, 45, 3, 40)	(63, 69, 5, 50)	(69, 73, 2, 50)	(69, 72, 0, 50)
7	(57, 63, 1, 40)	(63, 73, 1, 50)	(69, 76, 7, 50)	(73, 80, 4, 50)	(72, 78, 3, 50)

Źródło: Opracowanie własne

Tabela 3.17 NDCO dla przykładu z podrozdziału 3.4.

	0	1	2	3	4
0	(0, 10, 4, 10)	(0, 8, 2, 10)	(8, 12, 0, 20)	(0, 8, 0, 10)	(0, 9, 5, 10)
1	(10, 20, 2, 20)	(8, 16, 7, 10)	(20, 28, 2, 30)	(8, 13, 3, 10)	(9, 15, 6, 10)
2	(20, 28, 3, 20)	(16, 23, 4, 20)	(28, 33, 3, 30)	(19, 23, 1, 20)	(15, 19, 1, 10)
3	(28, 36, 7, 20)	(42, 45, 5, 30)	(33, 38, 4, 30)	(36, 46, 7, 30)	(28, 38, 2, 40)
4	(36, 42, 5, 20)	(45, 47, 0, 30)	(38, 40, 1, 30)	(46, 49, 6, 30)	(46, 49, 7, 40)
5	(42, 46, 6, 20)	(47, 54, 3, 40)	(49, 56, 7, 50)	(49, 55, 5, 40)	(49, 51, 4, 40)
6	(47, 53, 0, 40)	(54, 58, 6, 40)	(56, 62, 5, 50)	(55, 62, 4, 50)	(54, 60, 3, 50)
7	(53, 59, 1, 40)	(59, 69, 1, 50)	(62, 66, 6, 50)	(62, 66, 2, 50)	(60, 63, 0, 50)

Źródło: Opracowanie własne

Tabela 3.18 NTD dla przykładu z podrozdziału 3.4.

	0	1	2	3	4
0	(0, 10, 4, 10)	(0, 8, 7, 10)	(13, 17, 0, 20)	(0, 5, 3, 10)	(0, 9, 5, 10)
1	(10, 16, 5, 20)	(8, 16, 2, 10)	(23, 28, 4, 30)	(5, 13, 0, 10)	(9, 15, 6, 10)
2	(16, 24, 3, 20)	(16, 23, 4, 20)	(28, 33, 3, 30)	(19, 23, 1, 20)	(15, 19, 1, 10)
3	(24, 28, 6, 20)	(23, 26, 5, 30)	(33, 35, 1, 30)	(28, 31, 6, 30)	(28, 30, 4, 40)
4	(28, 36, 7, 20)	(26, 28, 0, 30)	(46, 54, 2, 30)	(36, 46, 7, 30)	(46, 49, 7, 40)
5	(36, 46, 2, 20)	(33, 40, 3, 40)	(54, 60, 5, 50)	(46, 52, 5, 40)	(54, 64, 2, 40)
6	(46, 52, 1, 40)	(40, 44, 6, 40)	(60, 64, 6, 50)	(52, 59, 4, 50)	(64, 70, 3, 50)
7	(52, 58, 0, 40)	(52, 62, 1, 50)	(64, 71, 7, 50)	(64, 68, 2, 50)	(70, 73, 0, 50)

Źródło: Opracowanie własne

Tabela 3.19 PPPO dla przykładu z podrozdziału 3.4

	0	1	2	3	4
0	(0, 10, 4, 10)	(0, 8, 2, 10)	(8, 12, 0, 20)	(0, 8, 0, 10)	(0, 9, 5, 10)
1	(10, 16, 5, 20)	(8, 16, 7, 10)	(30, 38, 2, 30)	(8, 13, 3, 10)	(9, 15, 6, 10)
2	(16, 20, 6, 20)	(16, 23, 4, 20)	(38, 40, 1, 30)	(19, 23, 1, 20)	(15, 19, 1, 10)
3	(20, 30, 2, 20)	(23, 26, 5, 30)	(40, 45, 4, 30)	(23, 26, 6, 30)	(38, 48, 2, 40)
4	(30, 38, 7, 20)	(26, 28, 0, 30)	(46, 51, 3, 30)	(38, 48, 7, 30)	(48, 50, 4, 40)
5	(38, 46, 3, 20)	(28, 32, 6, 40)	(54, 60, 5, 50)	(48, 54, 5, 40)	(50, 53, 7, 40)
6	(46, 52, 1, 40)	(51, 58, 3, 40)	(60, 64, 6, 50)	(54, 58, 2, 50)	(58, 61, 0, 50)
7	(52, 58, 0, 40)	(58, 68, 1, 50)	(64, 71, 7, 50)	(58, 65, 4, 50)	(61, 67, 3, 50)

Źródło: Opracowanie własne

Tabela 3.20 LOS dla przykładu z podrozdziału 3.4

	0	1	2	3	4
0	(0, 10, 4, 10)	(0, 8, 2, 10)	(8, 12, 0, 20)	(0, 8, 0, 10)	(0, 4, 1, 10)
1	(10, 20, 2, 20)	(8, 16, 7, 10)	(17, 19, 1, 30)	(8, 13, 3, 10)	(4, 13, 5, 10)
2	(20, 28, 3, 20)	(16, 23, 4, 20)	(20, 28, 2, 30)	(13, 17, 1, 20)	(13, 19, 6, 10)
3	(28, 34, 5, 20)	(23, 25, 0, 30)	(28, 33, 3, 30)	(38, 41, 6, 30)	(28, 38, 2, 40)
4	(34, 38, 6, 20)	(34, 37, 5, 30)	(33, 38, 4, 30)	(46, 56, 7, 30)	(38, 40, 4, 40)
5	(38, 46, 7, 20)	(37, 44, 3, 40)	(62, 68, 5, 50)	(56, 62, 5, 40)	(56, 59, 7, 40)
6	(46, 52, 0, 40)	(44, 48, 6, 40)	(68, 72, 6, 50)	(62, 66, 2, 50)	(59, 62, 0, 50)
7	(52, 58, 1, 40)	(58, 68, 1, 50)	(72, 79, 7, 50)	(66, 73, 4, 50)	(62, 68, 3, 50)

Źródło: Opracowanie własne

Analizując wygenerowane losowo dane oraz ustawione harmonogramy nie sposób jest doszukać się jakiegoś błędu. Pozostaje więc zastanowić się co sprawia, że wyniki otrzymane w pracy różnią się od tych założonych w części teoretycznej. Pierwsze co rzuca się w oczy to to, że na stanowisku numer 2 praca zaczyna się dopiero w terminie 8

lub 13 z powodu braku operacji numer 10 do wykonania na tym stanowisku. Z pewnością każde przedsiębiorstwo planując halę produkcyjną czy samą produkcję będzie dążyło do tego, aby praca na każdej maszynie mogła rozpocząć się już w terminie 0 – każde inne rozwiązanie to strata cennego czasu. Przykładowo na maszynie numer 4 nie występują w ogóle operacje numer 20 i 30. W związku z tym na stanowisko w zależności od reguły jest krótszy bądź dłuższy przestój (np. dla reguły NCO przestój ten stanowi niemal 31% całego harmonogramu). Widzimy więc, że wygenerowane losowo dane ni jak mają się do realnych, szczegółowo zaplanowanych systemów produkcyjnych.

3.4. Wnioski

Podsumowując wszystkie otrzymane w mojej pracy wyniki oczywiście jestem daleki od negowania przytoczonej wcześniej teorii. Różnica najprawdopodobniej wynika z braku danych z prawdziwych systemów wytwórczych. Dla losowo wygenerowanych danych, każda z pięciu reguł priorytetu szczegółowo omawianych w mojej pracy może układać najkorzystniejsze z punktu widzenia producenta harmonogramy. Metody NTD, PPPO oraz LOS dla 1000 przeliczonych zestawów danych uzyskały bardzo podobne do siebie wyniki, które są wynikami znacznie lepszymi od tych uzyskanych regułami NCO czy NTD. Skąd więc może wynikać przewaga tych trzech metod? Metodę NTD oraz PPPO łączy faworyzowanie od początku do końca tych samych detali, czy to ze względu na dyrektywny termin zakończenia czy kolejności napłyniętego zamówienia na detal. Pozostaje wytłumaczyć równie dobre rozwiązania reguły LOS. Tutaj się ponownie będę się posiłkował pracą Goli, Sobaszka i Świcia [5] – losowy algorytm potrafił w niej minimalizować wartości funkcji C_{max}, co sugeruje, że również dobre wyniki tej reguły otrzymane w mojej pracy nie są „przypadkowe”.

Podsumowanie

W pracy przedstawiono oraz porównano pięć wybranych prostych reguł priorytetu harmonogramowania produkcji. Narzędziem, które umożliwiło harmonogramowanie produkcji przez każdą regułę 1000 razy były stworzone samodzielnie algorytmy zaimplementowane w języku Python 3.9 przy pomocy oprogramowania PyCharm. Do porównania posłużyły otrzymane przez program czasy trwania harmonogramów, średnie czasy zakończenia obróbki poszczególnych detali oraz opóźnienia w stosunku do losowo wygenerowanych dyrektywnych terminów zakończenia obróbki poszczególnych elementów.

Zaskakująco okazało się, że najgorzej do spółki z regułą NDCO wypada reguła NCO. Wartości funkcji C_{max} dla tych reguł prezentowały się średnio na bardzo wyrównanym poziomie, jednak pod względem średniego czasu zakończenia obróbki poszczególnych detali, a co się z tym wiąże – mniejszymi opóźnieniami względem dyrektywnych terminów zakończenia obróbki charakteryzowała się reguła NCO.

Reguły NTD, PPPO oraz LOS średnio pozwoliły otrzymać niemal równe wyniki, z minimalną przewagą PPPO.

Zwłaszcza słabe wyniki proponowane przez regułę NCO kłócą się z literaturą zaprezentowaną w pracy. Pokazuje to jak istotne jest wcześniejsze zaplanowanie systemu produkcyjnego. Wyniki otrzymywane dla losowo generowanej produkcji wykazują wiele strat, które eliminuje się w produkcji rzeczywistej.

4. Załącznik

- Wczytanie niezbędnych bibliotek:

```python
import numpy as np
import random as random
import xlsxwriter
```

- Funkcja generująca losowe dane wejściowe oraz ustawiająca operacje rosnąco na poszczególnych stanowiskach, jest to zarazem uproszczona reguła LOS:

```python
def app(ziarno, metoda):
    random.seed(ziarno)
    operacja = [random.sample([10,20,30,40,50], 5) for i in range(8)]
    czas = [[random.randint(2, 10) for i in range(5)] for j in range(8)]
    termin = [random.randint(20, 30) for i in range(8)]
    kolejnosczamowienia = random.sample([0,1,2,3,4,5,6,7], 8)

    kolejnosc = np.zeros((8, 5)).astype(tuple)
    jakidetal = np.zeros((8, 5)).astype(int)
    poczatekikoniec = np.zeros((8, 5)).astype(tuple)
```

```python
    # ustawienie kolejności
    rzad = 0

    while rzad < 5:
        numer_operacji = 10
        detal = 0
        while detal < 8:

            for i in range(8):
                wartosc = operacja[i][rzad]

                if wartosc == numer_operacji:
                    kolejnosc[detal][rzad] = (i, rzad)
                    jakidetal[detal][rzad] = i
                    detal += 1

            numer_operacji += 10
        rzad+=1

    jakidetal, kolejnosc = metoda(kolejnosc, operacja, czas, jakidetal, kolejnosczamowienia, termin)
```

```python
    for j in range(5):
        for i in range(8):
            if i==0:
                poczatekikoniec[i][j]= (0, 0 + czas[kolejnosc[i][j][0]][kolejnosc[i][j][1]],
                                        jakidetal[i][j],
                                        operacja[kolejnosc[i][j][0]][kolejnosc[i][j][1]])
            elif i>0:
                poczatekikoniec[i][j]=(poczatekikoniec[i-1][j][1],
                                       poczatekikoniec[i-1][j][1]+czas[kolejnosc[i][j][0]][kolejnosc[i][j][1]],
                                       jakidetal[i][j],
                                       operacja[kolejnosc[i][j][0]][kolejnosc[i][j][1]])
```

```
    a=0
    b=(0,0,0)
    c=0

    x = -1 #nr detalu
    y = 0 #nr operacji
    while y < 40:
        y += 10
        x=-1
        while x < 7:
            x += 1
            for i in range(8):
                for j in range(5):
                    if (poczatekikoniec[i][j][2] == x) and (poczatekikoniec[i][j][3] == y):
                        a=poczatekikoniec[i][j][1]
                    if (poczatekikoniec[i][j][2]==x) and ((poczatekikoniec[i][j][3])==(y+10)):
                        b=(poczatekikoniec[i][j][0],i,j)
            if b[0]<a:
                for i in range(b[1],8):
                        poczatekikoniec[i][b[2]] = (poczatekikoniec[i][b[2]][0] + (a - b[0]),
                                                    poczatekikoniec[i][b[2]][1] + (a - b[0]),
                                                    poczatekikoniec[i][b[2]][2],
                                                    poczatekikoniec[i][b[2]][3]
                                                    )
```

```
    x=-1
    z=0
    while z < 10:
        z += 1
        for j in range(5):
            for i in range(7):
                if (poczatekikoniec[i][j][1]) != (poczatekikoniec[i+1][j][0]):
                    roznica = ((poczatekikoniec[i+1][j][0]) - (poczatekikoniec[i][j][1]))
                    a=poczatekikoniec[i+1][j][2]
                    b=((poczatekikoniec[i+1][j][3])-10)
                    for k in range(8):
                        for l in range(5):
                            if (poczatekikoniec[k][l][2] == a) and (poczatekikoniec[k][l][3] == b):
                                cofniecie = min(roznica,((poczatekikoniec[i+1][j][0])-(poczatekikoniec[k][l][1])))
                                poczatekikoniec[i+1][j] = (poczatekikoniec[i+1][j][0] - cofniecie,
                                                           poczatekikoniec[i+1][j][1] - cofniecie,
                                                           poczatekikoniec[i+1][j][2],
                                                           poczatekikoniec[i+1][j][3]
                                                           )

    koniec_linii = [poczatekikoniec[7][i][1] for i in range(5)]
    koniec_detalu =[]
```

```
    z = -1
    while z < 8:
        z += 1
        for j in range(5):
            for i in range(8):
                if ((poczatekikoniec[i][j][3] == 50) and (poczatekikoniec[i][j][2] == z)):
                    koniec_detalu.append(poczatekikoniec[i][j][1])

    opoznienie = [termin[i] - koniec_detalu[i] for i in range(8)]

    return (koniec_linii, koniec_detalu, opoznienie)
```

- Funkcja dla reguły NCO:

```
def nco(kolejnosc, operacja, czas, jakidetal, kolejnosczamowienia, termin):
    #ustawienie koleności dla NCO
    rzad = 0

    while rzad < 5:
        flag = True
        while flag:
            flag = False
            for i in range(8-1):
                wspolrzedne = kolejnosc[i][rzad]
                kolejny = kolejnosc[i+1][rzad]

                wartosc = operacja[wspolrzedne[0]][wspolrzedne[1]]
                kolejny_wartosc = operacja[kolejny[0]][kolejny[1]]
                if wartosc == kolejny_wartosc:
                    wartosc2 = czas[wspolrzedne[0]][wspolrzedne[1]]
                    kolejny_wartosc2 = czas[kolejny[0]][kolejny[1]]
                    if kolejny_wartosc2<wartosc2:
                        kolejnosc[i][rzad]=kolejny
                        detal1=jakidetal[i][rzad]
                        detal2=jakidetal[i+1][rzad]
                        jakidetal[i][rzad]=detal2
                        jakidetal[i+1][rzad]=detal1
                        kolejnosc[i+1][rzad] = wspolrzedne
                        flag = True
                        break
        rzad+=1
    return jakidetal, kolejnosc
```

- Funkcja dla reguły PPPO:

```
def pppo(kolejnosc, operacja, czas, jakidetal, kolejnosczamowienia, termin):
    rzad = 0
    while rzad < 5:
        flag = True
        while flag:
            flag = False
            for i in range(8-1):
                wspolrzedne = kolejnosc[i][rzad]
                kolejny = kolejnosc[i+1][rzad]
                wartosc = operacja[wspolrzedne[0]][wspolrzedne[1]]
                kolejny_wartosc = operacja[kolejny[0]][kolejny[1]]
                if wartosc == kolejny_wartosc:
                    wartosc2 = jakidetal[i][rzad]
                    kolejny_wartosc2 = jakidetal[i+1][rzad]
                    if kolejnosczamowienia[kolejny_wartosc2]<kolejnosczamowienia[wartosc2]:
                        kolejnosc[i][rzad] = kolejny
                        detal1 = jakidetal[i][rzad]
                        detal2 = jakidetal[i + 1][rzad]
                        jakidetal[i][rzad] = detal2
                        jakidetal[i + 1][rzad] = detal1
                        kolejnosc[i + 1][rzad] = wspolrzedne
                        flag = True
                        break
        rzad+=1
    return jakidetal, kolejnosc
```

- Funkcja dla reguły NTD:

```
def ntd(kolejnosc, operacja, czas, jakidetal, kolejnosczamowienia, termin):
    rzad = 0
    while rzad < 5:
        flag = True
        while flag:
            flag = False
            for i in range(8-1):
                wspolrzedne = kolejnosc[i][rzad]
                kolejny = kolejnosc[i+1][rzad]
                wartosc = operacja[wspolrzedne[0]][wspolrzedne[1]]
                kolejny_wartosc = operacja[kolejny[0]][kolejny[1]]
                if wartosc == kolejny_wartosc:
                    wartosc2 = jakidetal[i][rzad]
                    kolejny_wartosc2 = jakidetal[i+1][rzad]
                    if termin[kolejny_wartosc2]<termin[wartosc2]:
                        kolejnosc[i][rzad] = kolejny
                        detal1 = jakidetal[i][rzad]
                        detal2 = jakidetal[i + 1][rzad]
                        jakidetal[i][rzad] = detal2
                        jakidetal[i + 1][rzad] = detal1
                        kolejnosc[i + 1][rzad] = wspolrzedne
                        flag = True
                        break
        rzad+=1
    return jakidetal, kolejnosc
```

- Funkcja dla reguły NDCO:

```
def ndco(kolejnosc, operacja, czas, jakidetal, kolejnosczamowienia, termin):
    rzad = 0
    while rzad < 5:
        flag = True
        while flag:
            flag = False
            for i in range(8-1):
                wspolrzedne = kolejnosc[i][rzad]
                kolejny = kolejnosc[i+1][rzad]
                wartosc = operacja[wspolrzedne[0]][wspolrzedne[1]]
                kolejny_wartosc = operacja[kolejny[0]][kolejny[1]]
                if wartosc == kolejny_wartosc:
                    wartosc2 = czas[wspolrzedne[0]][wspolrzedne[1]]
                    kolejny_wartosc2 = czas[kolejny[0]][kolejny[1]]
                    if kolejny_wartosc2>wartosc2:
                        kolejnosc[i][rzad] = kolejny
                        detal1 = jakidetal[i][rzad]
                        detal2 = jakidetal[i + 1][rzad]
                        jakidetal[i][rzad] = detal2
                        jakidetal[i + 1][rzad] = detal1
                        kolejnosc[i + 1][rzad] = wspolrzedne
                        flag = True
                        break

        rzad+=1
    return jakidetal, kolejnosc
```

- Funkcja dla uproszczonej reguły LOS:

```
def los(kolejnosc, operacja, czas, jakidetal, kolejnosczamowienia, termin):
    return jakidetal, kolejnosc
```

- Zapis wyników do plików *.xlsx:

```
metody = [nco, ndco, pppo, ntd, los]
for metoda in metody:
    linie = []
    detale = []
    opoznienia = []
    print(metoda.__name__)
    for i in range(65,259):
        koniec_linii, koniec_detalu, opoznienie = app(i, metoda)
        linie.append(koniec_linii)
        detale.append(koniec_detalu)
        opoznienia.append(opoznienie)
    with xlsxwriter.Workbook(metoda.__name__ + '.xlsx') as workbook:
        worksheet = workbook.add_worksheet()

        for row_num, data in enumerate(linie):
            worksheet.write_row(row_num, 0, data)

        for row_num, data in enumerate(detale):
            worksheet.write_row(row_num, 6, data)

        for row_num, data in enumerate(opoznienia):
            worksheet.write_row(row_num, 15, data)
```

Spis Tabel

Spis Rysunków

Bibliografia

1. Dorota Burchart-Korol, Joanna Furman, *Zarządzanie produkcją i usługami,* Wydawnictwo Politechniki Śląskiej, Gliwice 2007.
2. Edward Pająk, *Zarządzanie produkcją – Produkt, technologia, organizacja,* Wydawnictwo Naukowe PWN SA, Warszawa 2006.
3. Marek Brzeziński, *Organizacja i sterowanie produkcją,* Agencja Wydawnicza PLACET, Warszawa 2002.
4. Mujanah Ezat, *Simulation of Production Scheduling in Manufacturing Systems,* Dublin City University, Dublin 1993.
5. Arkadiusz Gola, Łukasz Sobaszek, Antoni Świć, *Szeregowanie zadań produkcyjnych z uwzględnieniem dwuczynnikowej niepewności procesu,* Wydział Mechaniczny Politechnika Lubelska, Lublin 2017.
6. Łukasz Sobaszek, *Metody harmonogramowania produkcji,* Wydział Mechaniczny Politechnika Lubelska, Lublin 2011.
7. K. J.Wróblewski, R. Krawczyński, A. Kosieradzka, S. Kasprzyk, *Reguły priorytetu w sterowaniu przepływem produkcji*, WNT, Warszawa 1984.
8. Alan B. Downey, *Myśl w języku python! Nauka programowania,* Helion, Gliwice 2016
9. Mark Summerfield, *Python 3. Kompletne wprowadzenie do programowania. Wydanie II,* Helion, Warszawa 2010.
10. Tadeusz Sawik, *Badania operacyjne dla inżynierów zarządzania,* Wydawnictwo AGH, Kraków 1998.