Metody optymalizacji – laboratorium

zad. 0 Przeczytać opis pakietu `JuMP` (z języka `Julia`) w celu zapoznania się z możliwościami.

zad. 1 Tartak produkuje deski o standardowej szerokości 22 cali (każda deska ma ustaloną długość). Klienci firmy zamawiają jednak deski o mniejszej szerokości (i o tej samej długości, jak deski o standardowej szerokości). Aktualne zamówienia opiewają na 110 desek o szerokości 7 cali, 120 desek o szerokości 5 cali i 80 desek o szerokości 3 cali. Deski o mniejszej szerokości są odcinane z desek o standardowej szerokości. Na przykład firma może podjąć decyzję o przecięciu dużej deski na dwie deski po 7 cali i jedną deskę o szerokości 5 cali. W tym przypadku z deski standardowej tracona jest listwa o szerokości 3 cali. Firma chce wykonać zamówienie w ten sposób, aby zminimalizować ilość odpadów.

Wskazówka: Znajdź wszystkie możliwe sposoby podziału deski o szerokości 22 cali na deski o szerokościach 7, 5 i 3 cali. Przez x_i oznacz liczbę dużych desek rozcinanych i-tym sposobem.

Sformułować problem w postaci zadania programowania całkowitoliczbowego. Zapisać go korzystając z pakietu `JuMP` (z języka `Julia`) i rozwiązać wywołując solver `GLPK` (lub `Cbc`).

Oddzielić model od danych. Dane (szerokość standardowej deski, dane o zamówieniu) powinny być zadawane na ich podstawie powinien być generowany model. Program powinien generować możliwe sposoby podziału deski na podstawie danych o zamówieniu.

zad. 2 Dany jest zbiór zadań $J = \{1, \ldots, n\}$, które muszą być wykonane na jednej maszynie. Dla każdego zadania $j \in J$ dane są: czas potrzebny na wykonanie zadania p_j, waga tego zadania w_j oraz moment gotowości zadania r_j, t.j. moment r_j przed którym zadanie nie może być rozpoczęte.

Znaleźć harmonogram, który minimalizuje $\sum_{j \in J} w_j C_j$, gdzie C_j jest momentem zakończenia zadania j (momenty zakończenia nie są znane na wstępie i powinny być wyznaczone).

Sformułować problem w postaci zadania programowania całkowitoliczbowego. Zapisać go korzystając z pakietu `JuMP` (z języka `Julia`) i rozwiązać jakiś egzemplarz problemu wywołując solver `GLPK` (lub `Cbc`). Oddzielić model od danych. Dane (liczba zadań n, czasy potrzebne na wykonanie zadań p_j oraz momenty gotowości zadań r_j) powinny być zadawane. Maksymalnie sparametryzować zapis modelu.

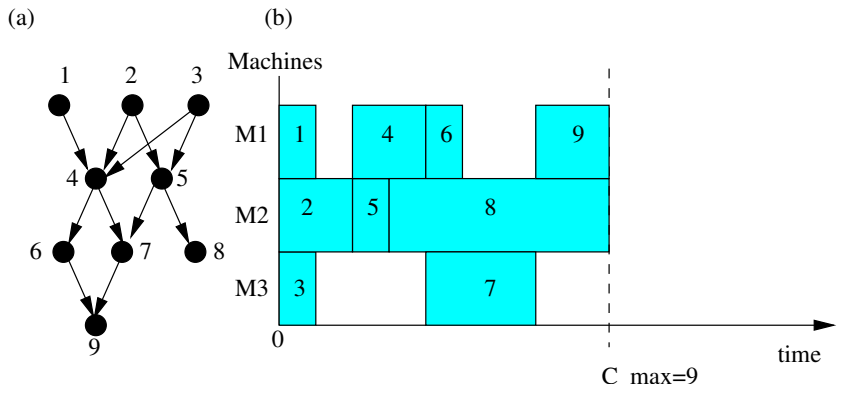
zad. 3 Dany jest zbiór zadań $J = \{1, \ldots, n\}$, który musi być wykonany na m maszynach. Dla każdego zadania $i \in J$ dany jest czas potrzebny na wykonanie zadania p_i. Zbiór zadań jest uporządkowany za pomocą relacji poprzedzania. Tzn. jeżeli $i \to j$, to zadanie j nie może się rozpocząć przed ukończeniem zadania i. Harmonogram jest dopuszczalny, jeśli spełnia ograniczenia poprzedzania.

Znaleźć dopuszczalny harmonogram, który minimalizuje całkowity czas potrzebny do wykonania wszystkich zadań oznaczony przez $C_{\max}$.

Rozważmy przykład podany na rysunku 1: liczba maszyn $m = 3$, liczba zadań $n = 9$, czasy wykonania $p_1 = 1$, $p_2 = 2$, $p_3 = 1$, $p_4 = 2$, $p_5 = 1$, $p_6 = 1$, $p_7 = 3$, $p_8 = 6$, $p_9 = 2$, relacje poprzedzania podane są na rysunku 1 a.

Rysunek 1 b pokazuje dopuszczalny harmonogram w stylu diagramu Gantt'a. $C_{\max} = 9$ dla tego harmonogramu.

Sformułować problem w postaci zadania programowania całkowitoliczbowego. Zapisać go korzystając z pakietu `JuMP` (z języka `Julia`) i rozwiązać jakiś egzemplarz problemu wywołując solver `GLPK` (lub `Cbc`). Oddzielić model od danych. Dane (liczba zadań n, liczba maszyn m, czasy potrzebne na wykonanie zadań p_j) powinny być zadawane. Maksymalnie sparametryzować zapis modelu. Program powinien wizualizować rozwiązanie na tekstowej konsoli w stylu diagramu. Taka wizualizacja pozwala łatwo sprawdzić dopuszczalność harmonogramu.



Rysunek 1: (a) Relacje poprzedzania. (b) Wizualizacja dopuszczalnego harmonogramu z całkowitym czasem potrzebnym do wykonania wszystkich zadań równym 9.

zad. 4 * Dany jest zbiór R złożony z p typów odnawialnych zasobów $R_1, R_2, \ldots, R_p$. Zasoby te są limitowane, tj. dla każdego R_i, $i = 1, \ldots, p$ podany jest limit N_i jednostek. Limity są stałe – nie zmieniają się w całym okresie planowania.

Dany jest zbiór czynności $Z = \{1, ..., n\}$. Dla każdej czynności $j \in Z$ dany jest czas jej wykonania t_j (w jednostkach czasowych) oraz wektor $\boldsymbol{r}_j = [r_1, r_2, \ldots, r_p]$ opisujący zapotrzebowanie na poszczególne zasoby $R_1, R_2, \ldots, R_p$, tzn. opisujący ilość jednostek zasobów zużywanych podczas wykonywania czynności j. Na czynności zbioru Z nałożone są ograniczenia kolejnościowe (Z jest częściowo uporządkowany). Ograniczenia kolejnościowe mogą być reprezentowane za pomocą grafu, w którym wierzchołki odpowiadają czynnością, a łuki określają poprzedzanie. Jeśli $k \to l$, to czynność l nie może być rozpoczęta przed ukończeniem czynności k.

Należy znaleźć harmonogram minimalizujący czas wykonania całego przedsięwzięcia. Harmonogram jest dopuszczalny jeśli spełnia ograniczenia kolejnościowe oraz przydział zasobów, zgodny z zapotrzebowaniem, nie przekracza podanych limitów w każdym momencie okresu planowania.

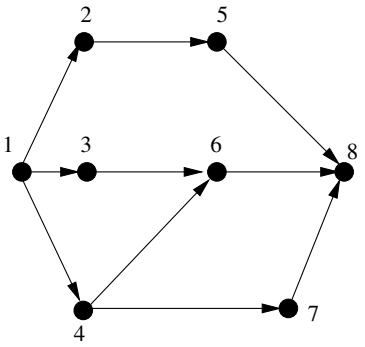
Sformułować problem w postaci zadania programowania całkowitoliczbowego. Zapisać go korzystając z pakietu `JuMP` (z języka `Julia`) i rozwiązać egzemplarz problemu (patrz poniżej) wywołując np. solver `GLPK` (lub `Cbc`). Oddzielić model od danych. Maksymalnie sparametryzować zapis modelu. Program powinien wizualizować rozwiązanie, np. na tekstowej konsoli, w stylu diagramu Gantt'a. Drukować również zapotrzebowanie na zasoby dla każdego momentu okresu planowania. Taka wizualizacja pozwala łatwo sprawdzić dopuszczalność harmonogramu.

Przykład egzemplarza problemu
Dane: liczba czynności $n = 8$, jeden typ zasobów (np. programiści) $p = 1$, limit zasobu $N_1 = 30$,

Czynność j	Czynności poprzedzające	Czasy wykonania t_j	Zapotrzeb. na zasoby $\boldsymbol{r}_j = [r_1]$
1	—	50	9
2	1	47	17
3	1	55	11
4	1	46	4
5	2	32	13
6	3,4	57	7
7	4	15	7
8	5,6,7	62	17

*Problem występuje podczas planowania i rozdziału zasobów np. w projekcie programistycznym.

Graf poniżej opisuje ograniczenia kolejnościowe.



Rozwiązania problemów przedstawić w sprawozdaniu, plik pdf, które powinno zawierać:

1. modele
   (a) definicje zmiennych decyzyjnych (opis, jednostki),
   (b) ograniczenia wraz z interpretacją (nie umieszczać źródeł modelu),
   (c) funkcje celu wraz z interpretacją,
2. wyniki oraz ich interpretację.

Model, zmienne w sprawozdaniu zapisujemy matematycznie (nie w języku `julia`) - zob. na stronie `przykład opisu modelu`.

Do sprawozdania należy dołączyć pliki w języku `julia` (`*.jl`). Pliki powinny być skomentowane: **imię i nazwisko** autora, komentarze zmiennych, zaetykietowane ograniczenia oraz komentarz ograniczeń.

Uwaga: Za zadania 1, 2, 3 (zadania obowiązkowe) można otrzymać co najwyżej ocenę dobrą. Zadania te będą punktowane następująco: zad. 1 - 10pkt, zad. 2 - 8pkt, zad. 3 - 12pkt i można otrzymać ocenę dst za 20pkt, dst+ za 25 pkt i db za 30pkt. Zad. 4 jest dodatkowe - jest na ocenę db+ lub bdb.