

ZASTOSOWANIE BADAŃ OPERACYJNYCH

Łukasz Indykiewicz (150142)
Karol Kozłowski (132652)
Stanisław Kozuchowicz (162655)
Karol Nikšcin (132750)
Łukasz Sierant (132823)
Mateusz Zarzyczny (132945)
Daniel Zasępa (132946)

1. Problem przepływu zadań.

Problem przepływowy jest zagadnieniem planowania produkcji: n zadań wymaga przetworzenia, w tej samej kolejności, na m maszynach. Czas wykonywania zadania i na maszynie j określony jest przez t_{ij} ($i = 1 \dots n$; $j = 1 \dots m$). Czasy wykonywania są ustalone, niezmiennie, nieujemne oraz niektóre z nich mogą być równe zero, gdy jakieś zadanie nie jest wykonywane na wybranej maszynie.

Zagadnienie sprowadza się do minimalizacji czasu pomiędzy rozpoczęciem pierwszego zadania na pierwszej maszynie, a zakończeniem ostatniego zadania na ostatniej maszynie. Czas ten w źródłach zwany jest „*makespan*”.

Przyjmuje się też następujące założenia:

- każde zadanie musi być przetworzone najwyżej raz na maszynie $1, 2, \dots, m$ (w tej kolejności);
- każda maszyna może przetwarzać jedno zadanie w tym samym czasie;
- każde zadanie może być przetwarzane na maksymalnie jednej maszynie w danej chwili czasu;
- zadania nie mogą być wywłaszczane;
- czasy rozpoczęcia operacji są wliczone do czasów operacji i nie zależą od sekwencji zadań;
- kolejność wykonywania zadań jest jednakowa na wszystkich maszynach.

Powyższy problem jest NP-trudny i może być dokładnie rozwiązany tylko dla małych rozmiarów. Sprowadza się do znalezienia sekwencji σ , która minimalizuje czas wykonania wszystkich zadań na wszystkich maszynach (*makespan*) $M(\sigma)$.

2. Porównanie wybranych algorytmów heurystycznych.

Wiele algorytmów zostało zaproponowanych do rozwiązania zagadnienia przepływowego.

W poniższej tabeli porównano jakość rozwiązań oraz złożoność obliczeniową części z nich, w którą wliczone jest również obliczanie *makespan*. Jakość rozwiązania podana jest w procentach powyżej średniej rozwiązań optymalnych (dla problemu 9 zadań, 10 maszyn).

Complexity		Quality					
Problems	–	500	100	100	100	50	50
Jobs	n	9	10	20	20	40	50
Machines	m	10	10	10	20	10	10
Gupta	$n \log(n) + nm$	13.4	12.8	19.6	18.8	18.9	17.1
Johnson	$n \log(n) + nm$	10.9	11.8	16.7	16.8	17.3	16.3
RA	$n \log(n) + nm$	8.5	9.1	12.5	13.4	13.5	11.2
Palmer	$n \log(n) + nm$	8.3	9.0	13.3	12.5	10.9	10.7
CDS	$nm^2 + nm \log(n)$	4.5	5.2	9.7	8.6	9.9	9.3
NEH	n^2m	2.1	2.2	3.9	3.8	2.6	2.1

Tabela 1: Zestawienie algorytmów heurystycznych.

NEH wydaje się być najlepszą heurystyką w praktyce. Heurystyki *RA* (*rapid access procedure*) lub *Palmer* mogą być także przydatne, gdy wymagane są krótkie czasy obliczeń. Złożoność NEH może zostać zredukowana z $O(n^3m)$ do $O(n^2m)$ – metoda ta została opisana w dalszej części tego tekstu.

3. Algorytm NEH.

- (1) Ułóż n zadań w malejącej kolejności sum czasów wykonania na maszynach.
- (2) Weź 2 pierwsze zadania i rozłóż je w takiej kolejności, aby zminimalizować, *makespan*.
- (3) Dla $k = 3$ do n powtarzaj krok (4).
- (4) Włóż k -te zadanie pomiędzy k dostępnych tak, aby *makespan* był możliwie najkrótszy.

Złożoność kroku (1) wynosi $O(n \log(n))$, kroku (2) $O(m)$. Wyliczenie jednego częściowego *makespanu* w kroku (4) zajmuje $O(km)$. Jednakże możliwe jest wyliczenie k *makespanów* w tego kroku w czasie $O(km)$.

4. Zmniejszenie złożoności algorytmu NEH.

Wyznaczamy M_i , *makespan* po ułożeniu zadania k na i -tej pozycji.

- (1) Oblicz najkrótszy czas ukończenia e_{ij} i -tego zadania na j -tej maszynie. Czas rozpoczęcia pierwszego zadania na pierwszej maszynie wynosi zero (Rysunek 1 a.).

$$e_{0j} = 0, \quad e_{i0} = 0,$$

$$e_{ij} = \max \{ e_{i,j-1}, e_{i-1,j} \} + t_{ij},$$

$$(i = 1, \dots, k-1), \quad (j = 1, \dots, m).$$

- (2) Oblicz ogon q_{ij} , tj. czas pomiędzy rozpoczęciem i -tego zadania na j -tej maszynie, a zakończeniem wszystkich zadań (Rysunek 1 b.).

$$q_{kj} = 0, \quad q_{i,m+1} = 0,$$

$$q_{ij} = \max \{ q_{i,j+1}, q_{i+1,j} \} + t_{ij},$$

$$(i = k-1, \dots, 1), \quad (j = m, \dots, 1).$$

- (3) Oblicz najwcześniejszy względnie czas ukończenia f_{ij} , na j -tej maszynie, zadania k ułożonego na i -tej pozycji (Rysunek 1 c.).

$$f_{i0} = 0,$$

$$f_{ij} = \max \{ f_{i,j-1}, e_{i-1,j} \} + t_{kj},$$

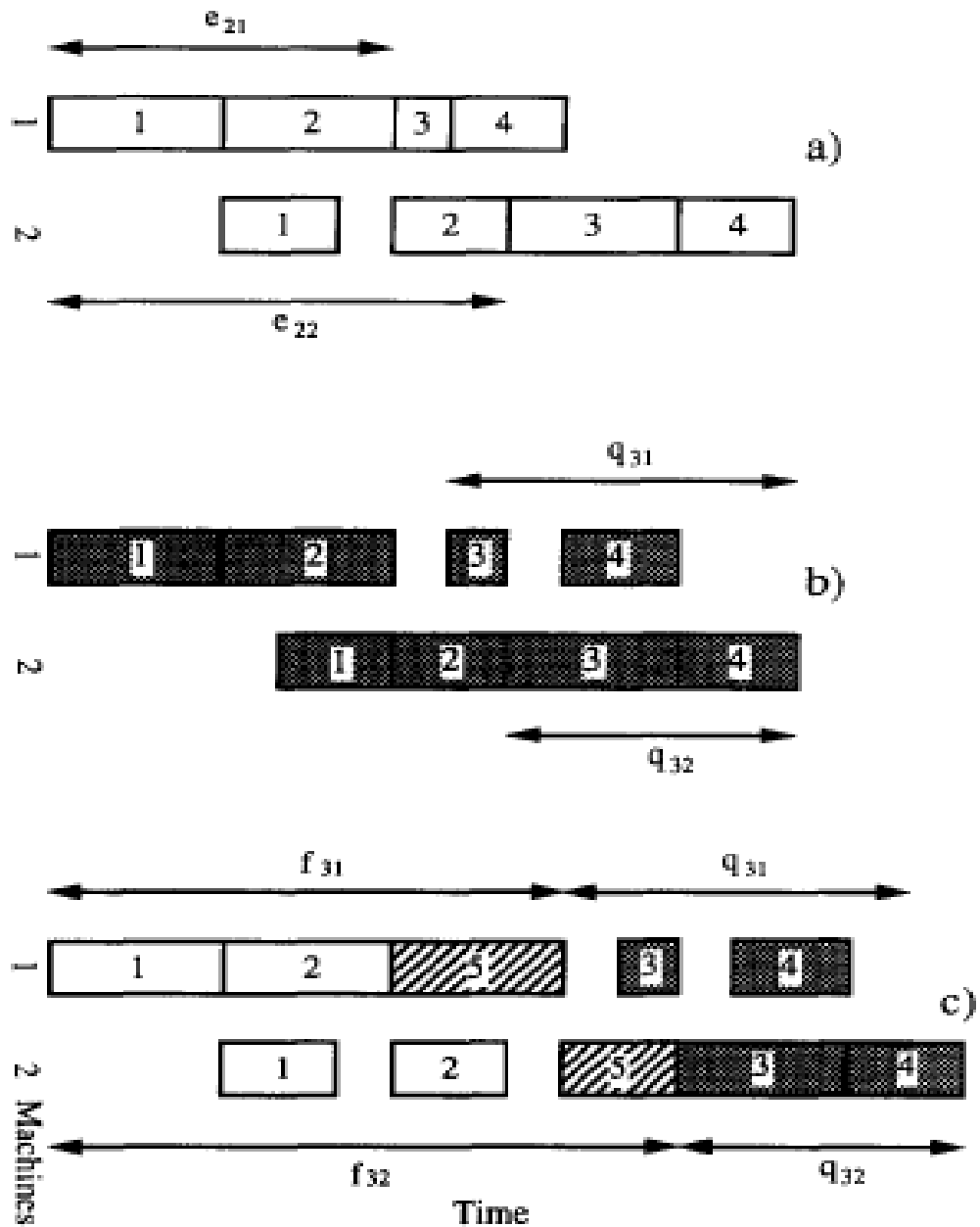
$$(i = 1, \dots, k), \quad (j = 1, \dots, m).$$

- (4) Wartość częściowego *makespanu* M_i po dodaniu k -tego zadania na i -tą pozycję wynosi:

$$M_i = \max_j (f_{ij} + q_{ij}),$$

$$(i = 1, \dots, k), \quad (j = 1, \dots, m).$$

Wszystkie te kroki wykonywane są w czasie $O(km)$. Zatem krok (4) algorytmu NEH ma złożoność obliczeniową $O(km)$. Więc cały algorytm NEH wykonuje się w czasie $O(n^2m)$.



Rysunek 1: Działanie algorytmu: wstawienie zadania nr 5 na trzecią pozycję.

5.Badania

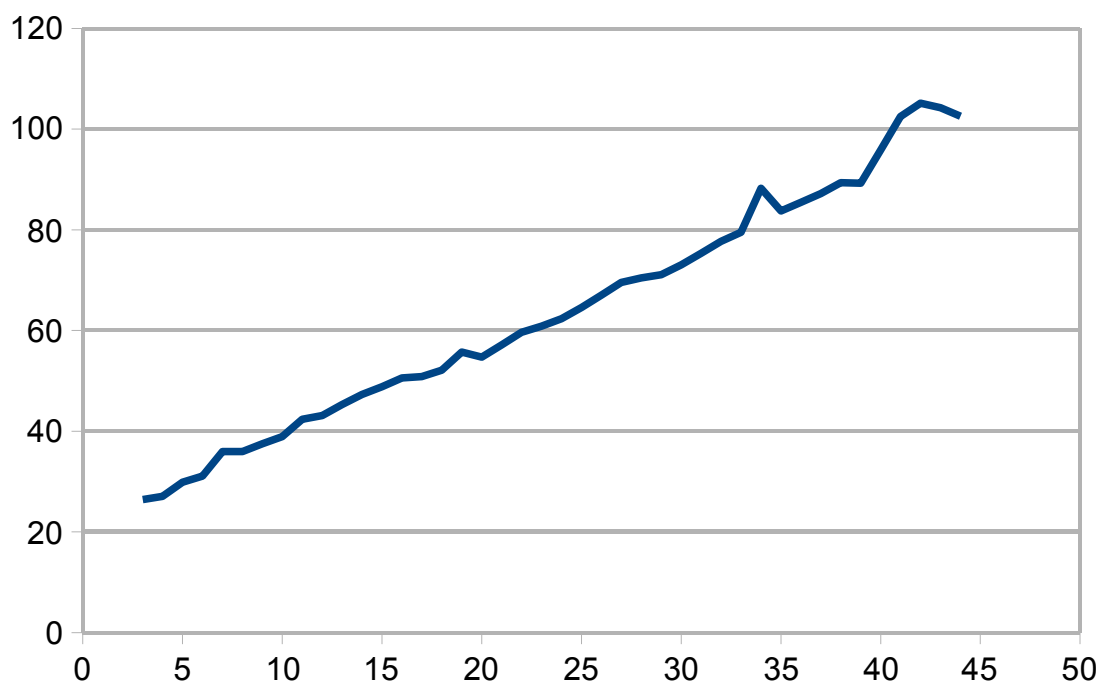
Wyniki badań są wartościami uśrednionymi z 500 wykonań algorytmu.

Czas wykonywania algorytmu mierzony był jako czas procesora potrzebny na wykonanie obliczeń i czynności dodatkowych takich jak wczytanie danych i inicjalizacja zmiennych.

a) wpływ liczby maszyn na prędkość algorytmu

Badanie polegało na sprawdzeniu wydajności algorytmu w zależności od ilości maszyn.

Dla celów badania przyjęto stałą liczbę 10 identycznych zadań o stałych czasach wykonywania na każdej maszynie. Zmianie ulegała natomiast liczba maszyn ($m \in [3, 45]$), na których wykonywane miały być zadania. Na poniższym wykresie (*Rysunek 2*) przedstawione są wyniki tych badań.

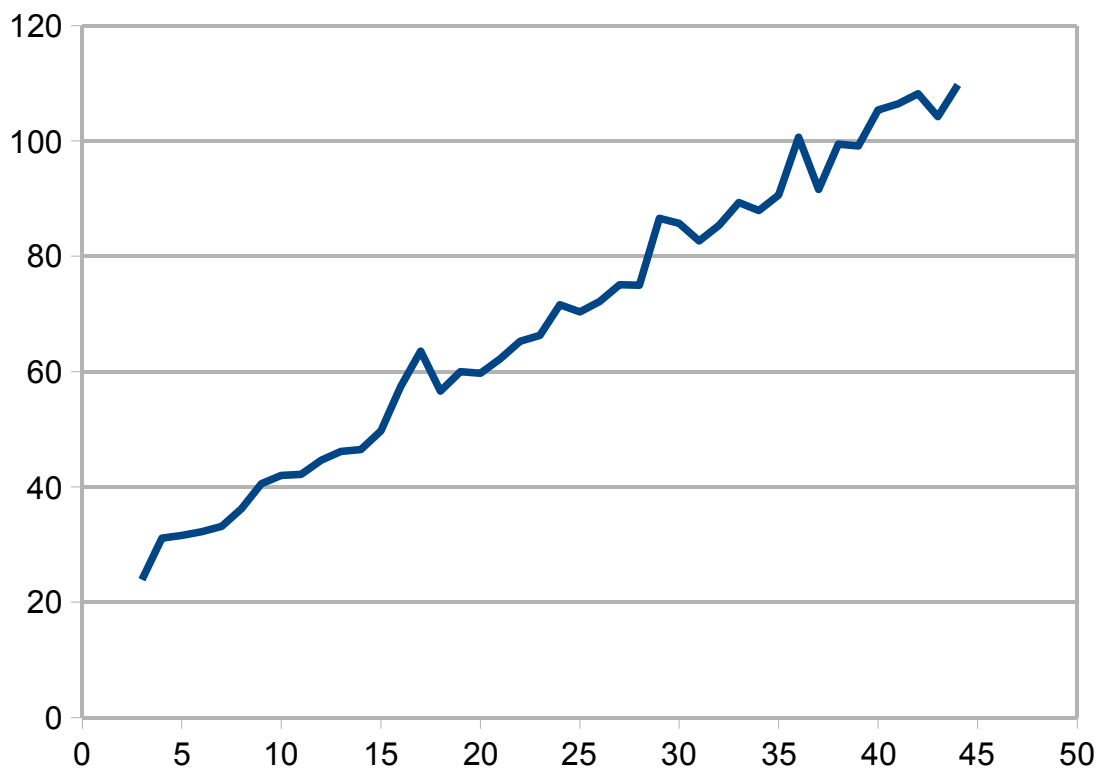


Rysunek 2: Wykres zależności czasu wykonania algorytmu od ilości maszyn

b) wpływ liczby zadań na prędkość algorytmu

Badanie polegało na sprawdzeniu wydajności algorytmu w zależności od ilości zadań.

Dla celów badania przyjęto stałą liczbę maszyn – 10 – na których wszystkie zadania były wykonywane w takim samym czasie. Zmianie ulegała natomiast liczba wykonywanych zadań ($n \in [3, 45]$). Na poniższym wykresie (*Rysunek 3*) przedstawione są wyniki tych badań.



Rysunek 3: Wykres zależności czasu wykonywania algorytmu od liczby zadań

6.Wnioski

Pomimo że złożoność obliczeniowa algorytmu wynosi $O(n^2m)$ badania pokazują liniową zależność czasu działania algorytmu od obydwu parametrów. Może się to wiązać z faktem, że algorytm działa bardzo szybko i większość mierzonego czasu poświęcana jest na pobranie danych.