Szymon DURAJ, Marek KUBALE, Tytus PIKIES
Politechnika Gdańska

# JAK TRANSPORTOWAĆ PRODUKTY CHEMICZNE, CZYLI PRZYPADEK WSADOWEGO SZEREGOWANIA ZADAŃ KOMPATYBILNYCH

**Streszczenie.** W pracy pokazano, że pewien problem transportu produktów chemicznych może być sprowadzony do problemu szeregowania identycznych zadań kompatybilnych na wsadowych maszynach jednorodnych i rozwiązany metodami kolorowania grafów. Ponieważ problem ten jest NP-trudny, zbadamy przypadki szczególne, które dają się rozwiązać w czasie $O(n^2)$. Rozważania ogólne będą wsparte doświadczeniami komputerowymi zebranymi w trakcie implementacji wybranych algorytmów szeregowania.

# HOW TO TRANSPORT CHEMICAL PRODUCTS, OR A SPECIAL CASE OF BATCH SCHEDULING OF COMPATIBLE JOBS

**Summary.** In the paper we show that a certain problem of transporting of chemical goods can be reduced to a problem of scheduling identical compatible jobs on parallel uniform batching machines and solved by means of graph coloring methods. Since the latter problem is generally NP-hard, we investigate some special cases that can be solved in time $O(n^2)$. Our theoretical considerations are accompanied by computer experiments conducted during the implementation of selected scheduling algorithms.

## 1. Wprowadzenie

Załóżmy, że musimy przesłać $n$ produktów chemicznych z miejsca $A$ do miejsca $B$ i mamy do dyspozycji 3 kontenery, które możemy wysłać transportem kolejowym, drogowym i/lub wodnym. Jednakże, w jednym kontenerze nie mogą znaleźć się takie produkty, które wchodzą ze sobą w reakcje (na przykład w wyniku zniszczenia pojemników podczas wypadku). Zadaniem jest takie przydzielenie pojemników chemicznych do kontenerów, by zminimalizować średni czas przewozu produktów, innymi słowy, aby suma czasów upływających od załadunku w $A$ do rozładunku w $B$ wszystkich produktów była jak najmniejsza.

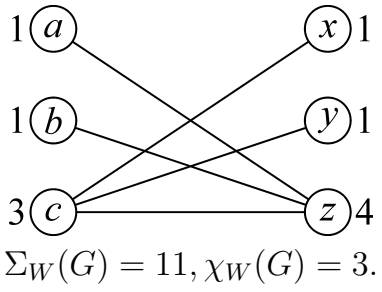
Zauważmy, że nasz problem może być wyrażony jako szczególny przypadek szeregowania zadań kompatybilnych na wsadowych maszynach jednorodnych typu $p - batch$ w celu minimalizacji średniego czasu przepływu. Ponieważ problem ten jest NP-trudny w przypadku ogólnym, przyjmiemy założenie, że zadania mają jednostkowy czas wykonywania, zaś graf niezgodności produktów jest dwudzielny. Sytuacja taka ma miejsce np. wówczas, gdy przewozimy cyjanki z kwasami. Symbolicznie, nasz pro-

blem szeregowania możemy zapisać jako $Qm|p-batch, p_j = 1, G = bipartite|\Sigma C_j$. Co prawda, założenie powyższe nie zmienia statusu złożoności problemu, ale ułatwia uzyskanie algorytmów wielomianowych w przypadkach szczególnych.

Powyższy problem szeregowania sprowadza się do kosztowego kolorowania grafów dwudzielnych, tj. takiego pokolorowania wierzchołków grafu niezgodności aby suma mocy poszczególnych kolorów pomnożonych przez wagi tych kolorów była jak najmniejsza. Wartość takiej sumy nazywamy *kosztową sumą chromatyczną grafu G*. W tym ujęciu:

- wierzchołki grafu są zadaniami,
- krawędzie oznaczają niemożność wykonania odpowiednich par zadań na jednej maszynie,
- kolory reprezentują maszyny,
- pokolorowanie danego wierzchołka oznacza przydzielenie danego zadania do odpowiedniej maszyny,
- wagi kolorów są odpowiednikami odwrotności szybkości maszyn,
- kosztowa suma chromatyczna jest minimalnym łącznym czasem przepływu zadań przez system.

Bardziej formalnie, niech $G$ będzie grafem $n$-wierzchołkowym i niech $W = (w_1, \ldots, w_n)$ będzie rosnącym ciągiem wag kolorów, gdzie $w_i$ jest liczbą rzeczywistą stowarzyszoną z kolorem numer $i$, $i = 1, ..., n$. *Kosztowa suma chromatyczna* związana z $W$ jest oznaczana jako $\Sigma_W(G)$, zaś *kosztowa liczba chromatyczna* $\chi_W(G)$ jest minimalną liczbą kolorów potrzebnych do osiągnięcia kosztowej sumy chromatycznej grafu $G$. Jeżeli $W$ jest ciągiem liczb naturalnych, czyli $N = (1, 2, 3, \ldots)$, to piszemy odpowiednio $\Sigma_N(G)$ i $\chi_N(G)$. W pracy skoncentrujemy się na grafach dwudzielnych $G = (V_1 \cup V_2, E)$. Maksymalny stopień wierzchołka takiego grafu oznaczymy przez $\Delta$. Jeśli $\Delta \leqslant 3$ ($\Delta \leqslant 4$), to grafy takie nazwiemy *podbikubicznymi* (*podbikwartycznymi*). Kubicka [6] pokazała, że dla każdego naturalnego $k \geqslant 2$ istnieje graf dwudzielny $G$ o odpowiednio wysokim stopniu $\Delta$, taki że $\chi_N(G) = k$. Z drugiej strony Kosowski [5] pokazał, że dla każdego podbikwartycznego $G$ mamy $\chi_N(G) \leqslant 3$. Najmniejszy graf $G$, dla którego $\chi_W(G) = 3$, jest pokazany na rys. 1. Problem optymalnego kolorowania kosztowego grafów dwudzielnych staje się NP-trudny począwszy od $\Delta = 5$ [7]. Jednakże dla grafów o $\Delta \leqslant 4$ problem może być rozwiązany w czasie wielomianowym [3].




Rys. 1. Przykład pokolorowania chromatycznego dla $W = (1, 3, 4, \ldots)$.

Z uwagi na NP-trudność problemu, w niniejszej pracy rozważymy dwa przypadki szczególne naszego zagadnienia szeregowania. W następnym punkcie rozważymy przypadek $m = 2$ i dwudzielne grafy ogólne. W punkcie 3 rozważymy przypadek $m = 3$ i grafy podbikwartyczne. W punkcie 4 przedstawimy doświadczenia komputerowe zebrane w trakcie implementacji wybranych algorytmów.

## 2. Przypadek $m = 2$

Niech $M_1$, $M_2$ będą dwiema maszynami typu $p - batch$ o szybkościach $s_1 \geqslant s_2$. Niech $w(S)$ oznacza sumę wag zadań odpowiadających wierzchołkom należącym do zbioru $S$. Zwróćmy uwagę, że tym razem ważymy zadania a nie kolory. Poniżej zmodyfikujemy algorytm z pracy [9] tak, aby konstruował dwupokolorowanie o maksymalnej różnicy wag między klasami.

**Algorytm 1** Uogólnione kolorowanie skrajnie niesprawiedliwe

**Wejście:** Dwudzielny graf $G$.
**Wyjście:** Dwupokolorowanie o największej szerokości kolorowania.
1: $Color_1 = Color_2 = \emptyset$.
2: Niech $G_1, ..., G_c$ będą składowymi spójności $G$.
3: **for** $i = 1$ to $c$ **do**
4: Niech $(C_1, C_2)$ będzie dwukolorowaniem $G_i$, takim że $w(C_1) \geqslant w(C_2)$.
5: $Color_1 = Color_1 \cup C_1$.
6: $Color_2 = Color_2 \cup C_2$.
7: **end for**
8: **return** $(Color_1, Color_2)$

Obecnie możemy wprowadzić nasz pierwszy algorytm szeregowania.

**Algorytm 2** Algorytm dla problemu $Q2|p - batch, p_j = 1, G = bipartite|\Sigma w_j C_j$

**Wejście:** Dwudzielny graf $G$.
**Wyjście:** Harmonogram optymalny względem kryterium $\Sigma w_j C_j$.
1: $(Color_1, Color_2) =$ *Uogólnione kolorowanie skrajnie niesprawiedliwe(G)*
2: $M_1 \leftarrow Color_1, M_2 \leftarrow Color_2$

**Lemat 1.** *Algorytm 2, o złożoności O($n^2$), jest optymalny dla problemu* $Q2|p - batch, p_j = 1, G = bipartite|\Sigma w_j C_j$.

*Dowód.* Napiszmy wzór na łączny ważony czas zakończenia zadań dla harmonogramu tworzonego przez ten algorytm. Niech $(V_1, V_2)$ będzie dowolnym dwukolorowanie $G$. Bez utraty ogólności załóżmy, że $V_1$ przydzielane jest do $M_1$, $V_2$ zaś do $M_2$. Skorzystajmy z równości $w(V(G)) = w(V_1) + w(V_2)$. Wówczas

$$\begin{aligned}\Sigma w_j C_j &= \frac{w(V_1)}{s_1} + \frac{w(V_2)}{s_2} = \frac{w(V_1)}{s_1} + \frac{w(V(G)) - w(V_1)}{s_2} \\ &= \frac{w(V_1)(s_2 - s_1)}{s_1 s_2} + \frac{w(V(G))}{s_2}.\end{aligned}$$

Na mocy tego, że $s_2 \leqslant s_1$, łatwo zauważyć, że przyporządkowanie zadań o maksymalnej łącznej wadze do $M_1$, czyli właśnie zadań odpowiadających $Color_1$, jest optymalne. Złożoność algorytmu 1 jest $O(n^2)$, toteż złożoność algorytmu 2 jest również $O(n^2)$. □

Zauważmy na marginesie, że przyjęcie ograniczenia na pojemność wsadu radykalnie zmienia status złożoności problemu już w przypadku jednomaszynowym. Rzeczywiście, ponieważ 3-kolorowanie sprawiedliwe grafów dwudzielnych z warunkiem $3|n$ jest NP-zupełne, więc przyjęcie rozmiaru wsadu równego $b = n/3$ powoduje, iż $C_{max} = 3/s$ jedynie wtedy, gdy graf niezgodności $G$ jest sprawiedliwie 3-barwny. Analogiczna obserwacja zachodzi przy kryterium $\Sigma C_j$.

**3. Przypadek** $m = 3$

W punkcie tym rozważymy przypadek, gdy $\Delta \leqslant 4$, symbolicznie $Q3|p - batch, p_j = 1, G = subbiquartic|\Sigma C_j$. Niech będzie dany graf podbikwartyczny $G = (V_1 \cup V_2, E)$, jego kopia $G^* = (V_1^* \cup V_2^*, E^*)$ oraz wagi kolorów $k_1 < k_2 < k_3$. Tworzymy sieć $D$ jako digraf $(V(D), A(D))$ z wierzchołkiem początkowym $s$ i końcowym $t$, w sposób następujący:

$$\begin{aligned} V(D) &= V(G^*) \cup V(G) \cup \{s\} \cup \{t\} \\ A(D) &= A_{1,2} \cup A_{2,1} \cup A_{s,1} \cup A_{2,t} \cup A_{1,1} \cup A_{2,2} \\ w(a) &= \begin{cases} 1 & \text{gdy } a \in A_{s_1} \cup A_{2,t} \\ k & \text{gdy } a \in A_{1,1} \cup A_{2,2} \\ \infty & \text{gdy } a \in A_{1,2} \cup A_{2,1} \end{cases} \end{aligned}$$

gdzie:

$$\begin{aligned} A_{1,2} &= \{(v_1, v_2) : v_1 \in V_1, v_2 \in V_2, \{v_1, v_2\} \in E(G)\} \\ A_{2,1} &= \{(v_2, v_1) : v_1 \in V_1^*, v_2 \in V_2^*, \{v_1, v_2\} \in E(G^*)\} \\ A_{s,1} &= \{s\} \times V_1 \\ A_{2,t} &= V_2 \times \{t\} \\ A_{1,1} &= \{(v_1^*, v_1) : v_1 \in V_1\} \\ A_{2,2} &= \{(v_2, v_2^*) : v_2 \in V_2\} \\ k &= (k_3 - k_2)/(k_2 - k_1) \end{aligned}$$

*Rozcięciem* $(S - T)$ sieci $D$ nazywamy taki podział zbioru wierzchołków na zbiory $S$ i $T$, że $s \in S$ a $t \in T$. Przez *pojemność* rozcięcia rozumiemy sumę wag łuków, których początek należy do $S$ a koniec do $T$. *Minimalne rozcięcie* jest rozcięciem o minimalnej pojemności. Niech $(S - T)$ będzie minimalnym rozcięciem sieci $D$. Dla $i = 1, 2$ definiujemy

$$S_i = V_i \cap S, \quad S_i^* = V_i^* \cap S, \quad T_i = V_i \cap T, \quad T_i^* = V_i^* \cap T$$

W pracy [3], która zawiera podstawy matematyczne przyjętego tu modelu, autorzy definiują związek pomiędzy rozcięciem sieci $D$ a pseudokolorowaniem grafu $G$. Zauważają również, że dla $k = (k_3 - k_2)/(k_2 - k_1)$, dzięki wyznaczeniu pewnego minimalnego

rozcięcia, można otrzymać optymalne w sensie kosztowej sumy chromatycznej pseudokolorowanie dla kolorów o wagach $(0, 1, 1 + k)$ które jest również optymalne dla kolorów o wagach $k_1, k_2, k_3$. Jak również, że metoda przedstawiona w pracy [3] pozwala przekształcić pseudokolorowanie w kolorowanie o nie większej liczbie kolorów niż $\lceil \Delta/2 \rceil + 1$ i o takiej samej lub mniejszej kosztowej sumie chromatycznej. Na tej podstawie możemy sformułować następujący algorytm szeregowania.

**Algorytm 3** Algorytm dla problemu $Q3|p - batch, p_j = 1, G = subbiquartic|\Sigma C_j$

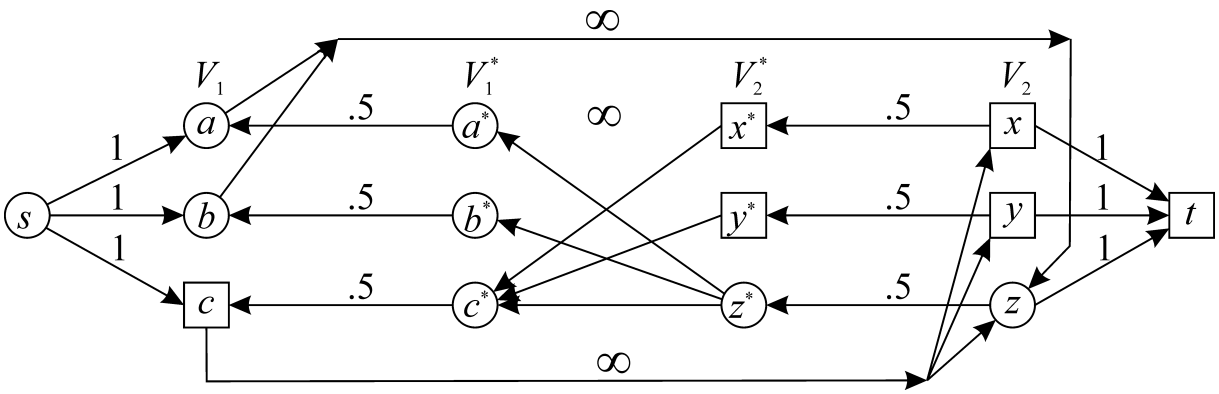
**Wejście:** Graf podbikwartyczny $G$, szybkości maszyn $s_1 > s_2 > s_3$
**Wyjście:** Uszeregowanie optymalne

1: Skonstruuj sieć $D$ dla $G$. Przyjmij $k = s_1(s_2 - s_3)/(s_3(s_1 - s_2))$.
2: Znajdź minimalne $(S, T)$-rozcięcie (np. za pomocą algorytmu Forda-Fulkersona [2]).
3: Zdefiniuj $Color_1 = S_1 \cup T_2, Color_2 = h^{-1}(T_1^* \cup S_2^*)$ i $Color_3 = V(G) \setminus (Color_1 \cup Color_2)$ [3].
4: Dokonaj korekty 3-pokolorowania grafu $G$ (szczegóły patrz [3]).
5: Przydziel $M_1 \leftarrow Color_1, M_2 \leftarrow Color_2, M_3 \leftarrow Color_3$.

Nasze wnioski można zawrzeć jako

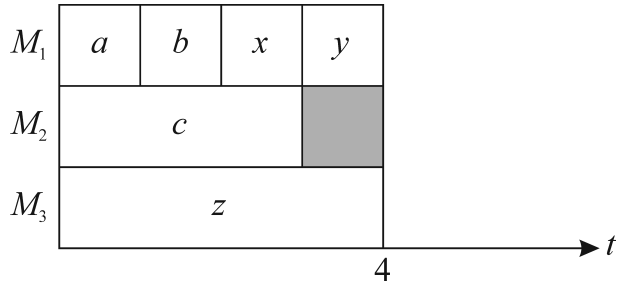
**Twierdzenie 1.** *Algorytm* 3 *rozwiązuje problem* $Q3|p - batch, p_j = 1, G = subbiquartic|\Sigma C_j$ *w czasie* $O(n^2)$.

*Dowód.* Znalezienie minimalnego rozcięcia w sieci $D$ można zrealizować w czasie $O(|A(D)||V(D)|)$ [4, 8]. W naszym przypadku $|A(D)| = 2|E(G)| + |V(G)| = O(n)$, zaś $|V(D)| = 2|V(G)| + 2 = O(n)$. Ponieważ pozostałe czynności algorytmu są liniowe, więc całkowita złożoność algorytmu wynosi $O(n^2)$. □




Rys. 2. Sieć $D$ dla grafu niezgodności z rys. 1. Wierzchołki okrągłe należą do zbioru $S$ - kwadratowe do $T$.

Dla kompletności rozważań zauważmy, że nasz problem staje się prosty, gdy $s_1 = s_2 \geqslant s_3$ oraz gdy $s_1 > s_2 = s_3$.




Rys. 3. Harmonogram optymalny dla grafu niezgodności z rys. 1.

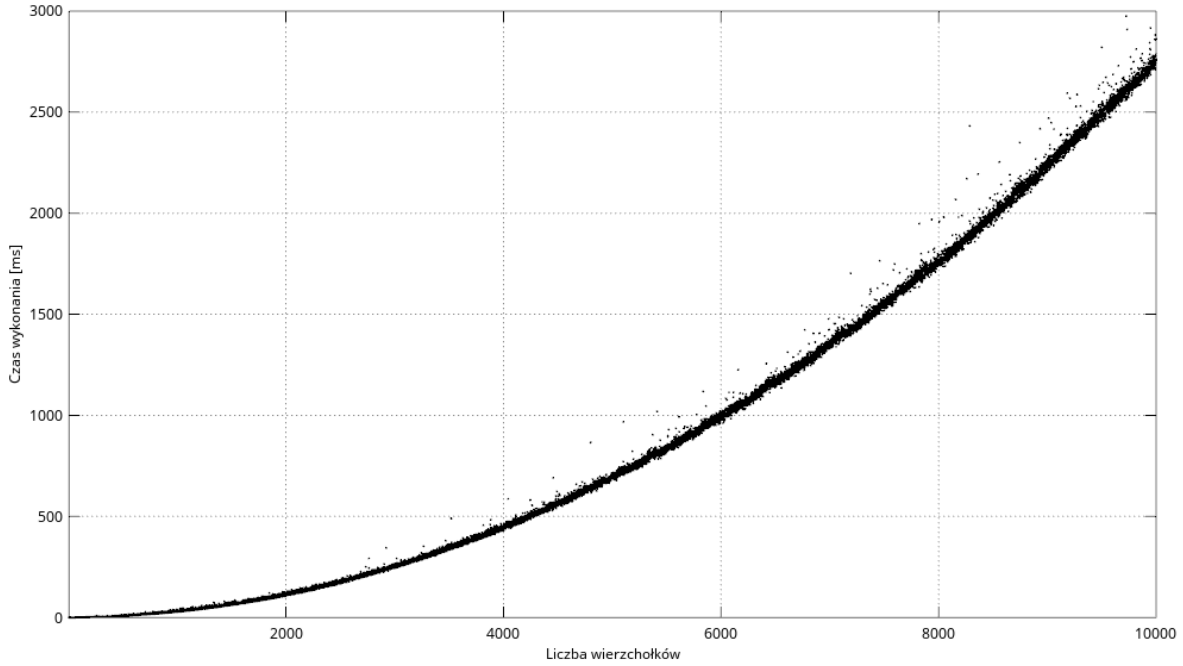
## 4. Eksperymenty komputerowe

Algorytm zaprezentowany w poprzednim punkcie został zaimplementowany w ramach pracy [1] i przetestowany na losowych grafach podbikubicznych. Wszystkie programy zostały napisane w języku C#. Użyto IDE Visual Studio Community 2015 (wersja 14.0.25431.01) z wykorzystaniem Microsoft .NET Framework w wersji 4.7.02053. Testy zostały wykonane na komputerze ASUS X555LB o procesorze Intel Core i5-5200U CPU 2.2 GHz. Podczas testów nie były uruchomiane żadne inne procesy poza procesami systemowymi. Testy zostały wykonane w jednym wątku.

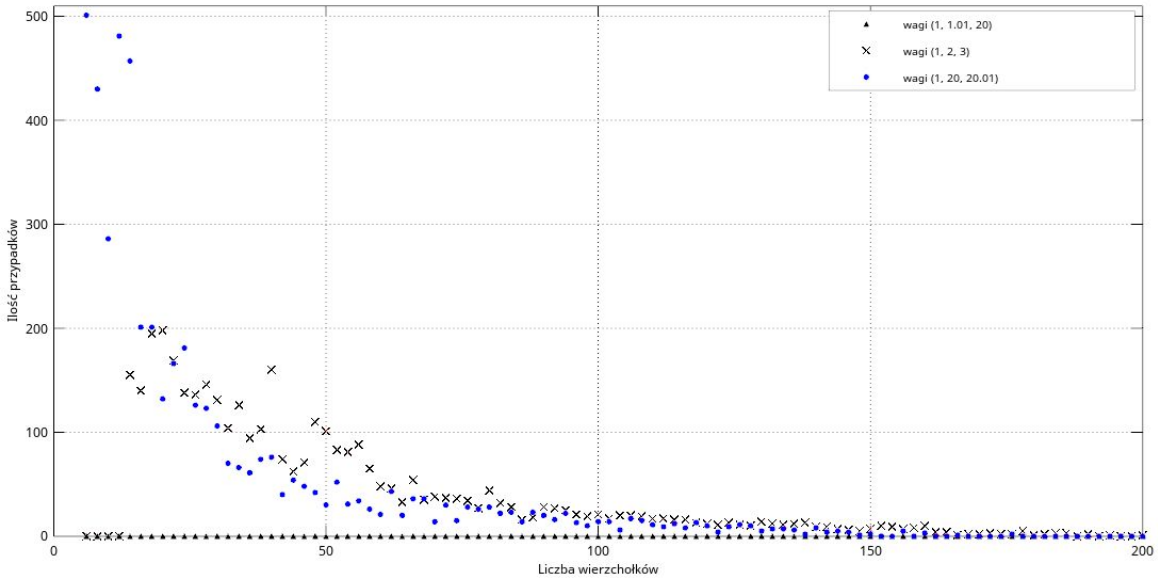
Najważniejszym był test czasu działania algorytmu w miarę zwiększania liczby węzłów. Spodziewaliśmy się wzrostu kwadratowego. Dla każdej wartości $n$ wygenerowano 3 spójne grafy podbikubiczne $n$-wierzchołkowe o równych bipartycjach, dokonano trzech pomiarów i uśredniono wyniki. Rys. 4 przedstawia wyniki pomiarów czasowych dla $n \in \{6, \ldots, 10000\}$. Jak widać, wyniki układają się w kształt połowy paraboli. Zauważono jednak, że w zdecydowanej większości (ponad 90% przypadków) do kolorowania niezbędne były jedynie 2 kolory. Jest to zrozumiałe, ponieważ przypadek użycia trzeciego koloru można uznać za szczególny. Aby odpowiedzieć na pytanie, jak często wymagane jest użycie trzeciego koloru, przygotowany został dodatkowy test, podczas którego 1000 razy generowany był spójny graf $n$-wierzchołkowy, $n \in \{6, \ldots, 200\}$, o równych bipartycjach, a następnie odnotowana została liczba grafów, do pokolorowania których wymagane było użycie dodatkowego koloru. Podobnych badań wykonano 50, po czym za ostateczny rezultat uznano maksymalny wynik. Oczywiście, na wyniki testów mają wpływ również wagi kolorów (a zatem szybkości maszyn). Zasadne wydało się zatem przetestowanie przypadków, gdy $k_1 = 2k_2 = 3k_3$, $k_1 << k_2 < k_3$ oraz $k_1 < k_2 << k_3$. Testy wykonano trzykrotnie dla następujących zestawów wag kolorów: $(1, 1.01, 20), (1, 2, 3), (1, 20, 20.01)$. Wyniki dla dwóch pierwszych przypadków zostały przedstawione na rys. 5. Wartości dla przypadku trzeciego są wszędzie równe 0.

Wyniki dla wszystkich przypadków tworzą widoczne linie trendu, które są funkcjami homograficznymi malejącymi wraz ze wzrostem $n$. Rezultaty są spodziewane - im większa jest suma kosztowa w przypadku dwóch kolorów (tj. większy jest rozmiar grafu), tym trudniej ją zmniejszyć przez dodanie trzeciego koloru. Częstość grafów, przy których jest to możliwe, spada wraz ze wzrostem liczby wierzchołków. Z eksperymentu wnioskujemy, że dla wartości $n$ powyżej 200 prawdopodobieństwo wylosowania takiego grafu jest bliskie zeru. Jednakże zauważmy, że na wartość kosztowej sumy chromatycznej wpływają głównie wagi kolorów. Przykładowo, nie istnieje taki graf 6-wierzchołkowy, którego kolorowanie dla wag $(1, 2, 3)$ wymagałoby użycia trzeciego

koloru. Prawdopodobnie najmniejszym takim grafem podbikubicznym jest nieukorzenione symetryczne drzewo binarne o 14 wierzchołkach.




Rys. 4. Czasy wykonania algorytmu 3 z podbikubicznymi grafami niezgodności w zależności od liczby wierzchołków grafu $G$.




Rys. 5. Liczba przypadków, w których konieczne było użycie trzeciego koloru.

LITERATURA

1. Duraj S.: Implementacja i testowanie algorytmów szeregowania zadań jednostkowych na procesorach jednorodnych (praca dyplomowa), WETI PG, Gdańsk (2017).
2. Ford L.R., Fulkerson D.R.: Maximal flow through a network, J. Canadien Math. 8 (1956), 399-404.
3. Giaro K., Kubale M.: Polynomial algorithm for cost coloring of bipartite graphs with $\Delta \leqslant 4$, Algorithmica (przedłożone do druku).
4. King V., Rao S., Tarjan R.: A faster deterministic maximum flow algorithm, J. of Algorithms 17 (1994), 447-474.
5. Kosowski A.: A note on the strength and minimum color sum of bipartite graphs, Disc. Appl. Math. 157 (2009), 2552-2554.
6. Kubicka E.: The Chromatic Sum of a Graph (praca doktorska), Western Michigan University (1989).
7. Małafiejski M., Giaro K., Janczewski R., Kubale M.: A 27/26-approximation algorithm for the chromatic sum coloring of bipartite graphs, Proc. APPROX'02, LNCS 2462 (2002), 136-146.
8. Orlin J.B.: Max flows in $O(nm)$ time, or better, Proc. STOC'13 (2013), 765-774.
9. Pikies T., Kubale M.: Better polynomial algorithms for scheduling unit-length jobs with bipartite incompatibility graphs on uniform machines, Bull. PAS - TS (w druku).