Teoria i praktyka wyznaczania najliczniejszego skojarzenia w grafach

Autoreferat rozprawy doktorskiej

Piotr Stańczyk
Instytut Informatyki UW

1 Zarys tematyki pracy

1.1 Sformułowanie problemu

Dla danego grafu $G = (V, E)$ *skojarzeniem* $M \subseteq E$ nazywamy taki podzbiór krawędzi grafu G, w którym żadne dwie krawędzie nie mają wspólnego końca. *Najliczniejszym skojarzeniem* nazywamy skojarzenie o największej liczności. *Skojarzenie doskonałe* to skojarzenie zawierające $|V|/2$ krawędzi. Problem najliczniejszego / doskonałego skojarzenia polega na znalezieniu najliczniejszego / doskonałego skojarzenia w danym grafie G.

1.2 Historia problemu

Problem najliczniejszego skojarzenia jest jednym z najczęściej badanych problemów grafowych — nie tylko ze względu na ciekawe podłoże teoretyczne, ale również ze względu na wiele praktycznych zastosowań, takich jak planowanie zadań czy zaawansowane metody obróbki obrazu. Historię algorytmów rozwiązujących problem skojarzenia przedstawiamy w dwóch, niezależnych odsłonach — sekwencyjnej i równoległej. W dalszej części przez n będziemy oznaczali liczbę wierzchołków, a przez m liczbę krawędzi grafu.

1.2.1 Algorytmy sekwencyjne

Historia algorytmów sekwencyjnych sięga końca XIX wieku, kiedy to Petersen wydał bardzo ważną publikację na temat teorii skojarzeń zatytułowaną „Die Theorem der regulären Graphs" [24]. Petersen udowodnił, że każdy kubiczny graf bez mostów posiada doskonałe skojarzenie. Dowód twierdzenia Petersena przeprowadzony pzez Frinke'a [12] w 1926 roku dostarcza jednocześnie algorytmu wyznaczającego takie skojarzenie w czasie $O(n^2)$.

Problem znajdowania najliczniejszego skojarzenia w dowolnych grafach czekał dziesięciolecia na rozwiązanie w czasie wielomianowym. Z czasem okazało się, że

znacznie łatwiej rozwiązać problem dla grafów dwudzielnych. Charakterystyka najliczniejszych skojarzeń w grafach dwudzielnych została podana w 1931 roku niezależnie przez König'a [19] i Egerváry'ego [9]. Ich konstruktywne dowody dostarczyły wielomianowego algorytmu — tak zwanej metody węgierskiej. Ponad trzydzieści lat później Edmonds [8] zaprezentował pierwszy wielomianowy algorytm dla dowolnych grafów. Działający w czasie $O(n^4)$ algorytm, jak się później okazało, był daleki od rozwiązania optymalnego. W 1976 roku Gabow [13] zoptymalizował algorytm Edmonds'a redukując jego złożoność do $O(n^3)$. W międzyczasie, w roku 1973, Hopcroft i Karp [17] zaprezentowali algorytm dla grafów dwudzielnych działający w czasie $O(m\sqrt{n})$. Siedem lat później Micali i Vazirani [21] osiągnęli tę samą złożoność dla dowolnych grafów. Wiele innych algorytmów zostało zaproponowanych w latach późniejszych:

- 1991, Alt, Blum, Mehlhorn i Paul [14] — algorytm dla grafów dwudzielnych działający w czasie $O(n^{3/2}\sqrt{m/\log n})$.
- 1991, Feder i Motwani [10] — algorytm dla grafów dwudzielnych działający w czasie $O(\frac{\sqrt{n}\,m\,\log(n^2/m)}{\log n})$.
- 2003, Fremuth-Paeger i Jungnickel [11] — uogólnienie algorytmu Feder'a i Motwani'ego na dowolne grafy.
- 2004, Mucha i Sankowski [23] — randomizowany algorytm dla dowolnych grafów działający w czasie $O(n^\omega)$ [1].
- 2006, Harvey [15] — uproszczony algorytm Muchy i Sankowskiego.

Z teoretycznego punktu widzenia najszybszym algorytmem do wyznaczania najliczniejszego skojarzenia w dowolnych grafach gęstych jest algorytm Muchy i Sankowskiego [23]. Jednakże, ze względu na wykorzystywanie przez ten algorytm skomplikowanej metody szybkiego mnożenia macierzy, jest on niepraktyczny.

Pomimo wieloletnich starań okazało się, że stworzenie efektywnego algorytmu dla problemu najliczniejszych skojarzeń w ogólnym przypadku jest bardzo trudne. Dlatego też przez lata koncentrowano się na poszukiwaniu rozwiązań dla różnych klas grafów. Przykładem mogą być choćby grafy planarne. Dzięki zastosowaniu eliminacji Gaussa autorstwa Lipton'a, Rose'a i Tarjan'a (tzw. algorytmu *nested dissection*) Mucha i Sankowski [22] zmodyfikowali swój algorytm i uzyskali złożoność $O(n^{\omega/2})$ — jest to, z punktu widzenia teorii, najszybszy algorytm dla grafów planarnych.

Inną interesującą klasą grafów są grafy kubiczne. Cieszą się one zainteresowaniem od dość niedawna. Wszystko dzięki Biedl [3], która zaprezentowała liniową redukcję problemu najliczniejszego skojarzenia w dowolnym grafie do problemu najliczniejszego skojarzenia w grafie kubicznym. Wynik ten pozwala zamienić dowolny wielomianowy algorytm dla grafów kubicznych o złożoności $O(f(n,m))$, w algorytm dla dowolnych grafów działający w czasie $O(f(n,m)+m)$. Grafy kubiczne są najprostszą klasą grafów, dla których problem skojarzeń jest tak samo trudny jak problem ogólny.

[1] $O(n^\omega)$ to czas optymalnego mnożenia macierzy, $\omega < 2.376$.

Ograniczając się do analizy różnych podklas grafów można uzyskać algorytmy szybsze od opisanych do tej pory. Przykładem może być choćby wspomniany dowód twierdzenia Petersen'a [24] zaproponowany przez Frinke'a. W 2001 roku Biedl i inni [4] zaprezentowali nowy algorytm do konstruowania doskonałych skojarzeń w grafach kubicznych bez mostów działający w czasie $O(n \log^4 n)$ (grafy takie zawsze posiadają doskonałe skojarzenie). Innymi algorytmami tego typu są: liniowy algorytm dla grafów kubicznych planarnych [4], czy też algorytm Schrijver'a [25] dla grafów kubicznych dwudzielnych.

1.2.2 Algorytmy równoległe

Problem równoległego wyznaczania najliczniejszego skojarzenia w dowolnych grafach jest co najmniej tak ciekawy, jak sekwencyjna wersja tego zagadnienia. Pytanie — czy można wyznaczyć najliczeniejsze skojarzenie w dowolnym grafie w czasie polilogarytmicznym, pozostaje bez odpowiedzi. Wprawdzie istnieją równoległe algorytmy randomizowane rozwiązujące ten problem, jednak nikomu do tej pory nie udało się zaprezentować algorytmu deterministycznego, czy też udowodnić, że taki algorytm nie istnieje.

Wzmożone badania nad algorytmami równoległymi rozpoczęły się około 40 lat temu, ale główna teoria wykorzystywana przez równoległe algorytmy algebraiczne była znana już w latach 40-tych XX wieku. W 1947 roku Tutte [30] stwierdził, że graf posiada doskonałe skojarzenie wtedy i tylko wtedy, gdy wyznacznik symbolicznej macierzy sąsiedztwa grafu jest różny od zera. W 1982 roku Borodin, Gathen i Hopcroft [5] zaprezentowali, bazując na twierdzeniu Tutte'a i na fakcie, że można wyliczyć wyznacznik numerycznej macierzy równolegle, randomizowany algorytm równoległy weryfikujący istnienie doskonałego skojarzenia w dowolnym grafie w czasie $O(\log^2 n)$.

W 1986 roku Karp, Upfal i Widgerson [18] opublikowali artykuł p.t. „Constructing a perfect matching is in Random NC", w którym zaprezentowali pierwszy równoległy algorytm konstruujący doskonałe skojarzenie. Działa on w czasie $O(\log^3 m)$ i wykorzystuje $O(n^{6.5})$ procesorów.

W 1988 roku Moitra i Johnson [2] zaproponowali algorytm dla grafów odcinkowych działający w czasie $O(\log^2 n)$ i używający $O(n^6/\log n)$ procesorów. W 1989 roku Dahlhaus, Karpinski i Lingas [7] analizowali planarne grafy dwudzielne. Zaproponowali oni algorytm działający w czasie $O((n/2 - l + \sqrt{n}) \log^7 n)$ na $O(n^{1.5} \log^3 n)$ procesorach (l to wielkość najliczniejszego skojarzenia w grafie). Bardziej ogólna klasa grafów — grafy dwudzielne, zostały zanalizowane przez Chaudhuri [6] w 1994 roku. Skonstruował on algorytm działający w czasie $O(n \log \log^2 n)$ na $O(n^3/\log \log n)$ procesorach.

W 1996 Sharan i Wigderson [27] zaprezentowali nowe podejście do konstruowania doskonałego skojarzenia. Uzyskali oni algorytm dla dwudzielnych grafów kubicznych działający w czasie $O(log^2 n)$ i wykorzystujący $O(n \log^* n/\log n)$ procesorów.

2 Ważne wyniki

Zanim przejdę do prezentacji wyników uzyskanych w przedłożonej pracy, pokrótce przedstawię trzy algorytmy, których znajomość jest bardzo pomocna w ich zrozumieniu. Dokładniejszy opis tych algorytmów zaprezentowany został w rozprawie doktorskiej.

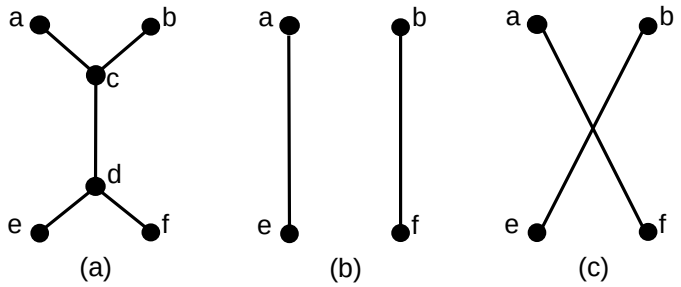
2.1 Algorytm Frinke'a

Algorytm Frinke'a do wyznaczania doskonałego skojarzenia w grafach kubicznych bez mostów w czasie $O(n^2)$ jest bezpośrednim rezultatem dowodu twierdzenia Petersena zaprezentowanego przez Frinke'a.

Teoria 1 (*Petersen*)**.** *Niech $G = (V, E)$ będzie grafem kubicznym nie zawierającym mostów. Wówczas G posiada doskonałe skojarzenie.*

Algorytm Frinke'a korzysta z obserwacji, zgodnie z którą graf kubiczny bez mostów można zredukować do grafu o dwa wierzchołki mniejszego poprzez usunięcie sąsiadujących wierzchołków c i d i połączenie wierzchołków a, b, e i f w jeden z dwóch sposobów zaprezentowanych na rysunku 1 (mogą występować przypadki szczególne, w których niektóre krawędzie są podwójne — dokładna analiza wszystkich przypadków została zaprezentowana w rozprawie doktorskiej). Frink udowodnił, że co najmniej jedna z tych dwóch redukcji prowadzi do grafu bez mostów. Wykonując sekwencję odpowiednich redukcji uzyskujemy w końcu graf pusty, w którym doskonałe skojarzenie M też jest puste. Następną fazą algorytmu Frinke'a jest wycofanie redukcji z jednoczesną aktualizacją konstruowanego doskonałego skojarzenia. Proces wycofywania redukcji prezentuje rysunek 2.

Ponieważ algorytm Frinke'a wykonuje $O(n)$ faz redukcji, a każda z nich wymaga sprawdzenia, czy uzyskany graf nie posiada mostów (co można zrealizować w czasie liniowym), cały algorytm działa w czasie $O(n^2)$. Dokładny opis algorytmu został zaprezentowany w rozdziale 4.1 rozprawy.



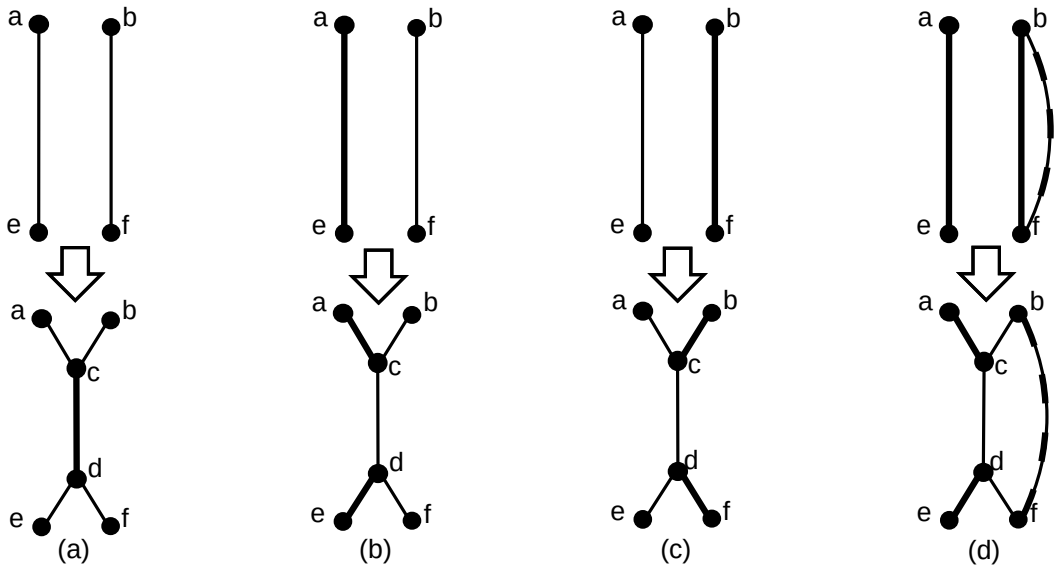
Rysunek 1: Redukcja grafu kubicznego bez mostów względem krawędzi $c - d$ zaproponowana przez Frinke'a. **(a)** Wierzchołki c i d zostają usunięte z grafu. **(b)** Pierwszy rodzaj redukcji — wierzchołek a jest połączony z e, wierzchołek b jest połączony z f. **(c)** Drugi rodzaj redukcji — wierzchołek a jest połączony z f, wierzchołek b jest połączony z e.

2.2 Algorytm Biedl

Biedl i inni [4] zaproponowali modyfikację algorytmu Frinke'a o złożoności $O(n \log^4 n)$. W celu uzyskania lepszej złożoności niż $O(n^2)$ konieczne jest przyspieszenie dwóch elementów algorytmu — poszukiwania mostów w grafie oraz poszukiwania cyklu naprzemiennego, który jest potrzebny w realizacji redukcji z rysunku 2(d). Pierwszy z problemów może zostać rozwiązany przez zastosowanie dynamicznych struktur danych dla problemu dwuspójności grafu (każdy graf kubiczny bez mostów jest również dwuspójny). Biedl zastosowała strukturę danych zaprezentowaną w [16], której zamortyzowany czas działania pojedynczej operacji wynosi $O(\log^4 n)$. Drugi z problemów został wyeliminowany dzięki następującej obserwacji:

Lemat 1. *Dla dowolnej krawędzi p dwuspójnego grafu kubicznego G istnieje doskonałe skojarzenie nie zawierające p.*

Algorytm wybiera dowolną krawędź p i dokonuje kolejnych redukcji Frinke'a względem krawędzi sąsiedniej z p (każda redukcja powoduje modyfikację krawędzi p — przykładowo krawędź $a - c$ na rysunku 1(a) staje się krawędzią $a - e$ na rysunku 1(b)). Następnie konstruowane jest początkowe skojarzenie nie zawierające krawędzi



Rysunek 2: Cztery możliwe przypadki wycofywania redukcji Frinke'a. **(a)** Żadna z usuwanych z grafu krawędzi nie jest skojarzona — dodaj krawędź $c-d$ do skojarzenia. **(b)** Krawędź $a - e$ należy do skojarzenia — usuń $a - e$ ze skojarzenia i dodaj $a - c$ i $d - e$ do skojarzenia. **(c)** Krawędź $b - f$ należy do skojarzenia — usuń $b - f$ ze skojarzenia i dodaj $b - c$ i $d - f$ do skojarzenia. **(d)** Krawędzie $a - e$ i $b - f$ są w skojarzeniu — znajdź cykl naprzemienny C zawierający co najmniej jedną z tych krawędzi (w przykładzie $c - b - \cdots - f - d$) i zmień przynależność w skojarzeniu wszystkich krawędzi z C. W ten sposób przypadek (d) jest redukowany do jednego z (a), (b) lub (c).

p, dzięki czemu faza wycofywania redukcji nie natrafia na przypadek z rysunku 2(d). Szczegółowa prezentacja algorytmu została zawarta w rozdziale 4.2 rozprawy.

2.3 Algorytm Muchy-Sankowskiego

Zaproponowany w 2004 roku algorytm Muchy-Sankowskiego [20] to najszybsza metoda wyznaczania maksymalnego skojarzenia w dowolnych grafach gęstych. Algorytm ten działa w czasie $O(n^\omega)$ i wykorzystuje metody algebraiczne. Dla danego grafu $G = (V, E)$, gdzie $n = |V|$, konstruowana jest symboliczna macierz sąsiedztwa T o wymiarach $n \times n$ zdefiniowana następująco:

$$T[x,y] = \begin{cases} 0 & \text{jeśli } V[x] - V[y] \notin E \\ e_{x,y} & \text{jeśli } V[x] - V[y] \in E \text{ i } x < y \\ -e_{y,x} & \text{jeśli } V[x] - V[y] \in E \text{ i } x > y \end{cases}$$

gdzie $e_{x,y}$ to zmienne symboliczne. Okazuje się, że wyznacznik symbolicznej macierzy T jest różny od zera wtedy i tylko wtedy, gdy G posiada doskonałe skojarzenie. Wyliczenie wyznacznika macierzy symbolicznej jest bardzo kosztowne, ale jeśli zmienne symboliczne zastąpi się losowymi elementami ciała Z_p, dla odpowiednio dużej liczby pierwszej p (otrzymując w ten sposób macierz numeryczną T'), to zerowanie się wyznacznika macierzy T' jest równoważne nieistnieniu doskonałego skojarzenia w grafie z dużym prawdopodobieństwem.

Jeśli macierz T' jest osobliwa, to graf prawdopodobnie nie posiada doskonałego skojarzenia. Aby w takiej sytuacji wyznaczyć wielkość maksymalnego skojarzenia, wystarczy znaleźć maksymalną nieosobliwą podmacierz T', która odpowiada podgrafowi grafu G posiadającemu doskonałe skojarzenie.

W celu skonstruowania doskonałego skojarzenia, algorytm Muchy-Sankowskiego testuje kolejne krawędzie grafu, sprawdzając czy są one dozwolone (należą do pewnego doskonałego skojarzenia) i jeśli tak, to dodaje je do konstruowanego wyniku. Testu krawędzi pod względem bycia dozwoloną można dokonać poprzez usunięcie z macierzy T' wierszy i kolumn odpowiadających badanej krawędzi i sprawdzenie, czy uzyskana w ten sposób macierz ma wyznacznik różny od zera.

Stosując zależność między wyznacznikami podmacierzy macierzy T' a macierzą T'^{-1} nie jest konieczne liczenie wyznacznika za każdym razem:

$$T'^{-1}[x,y] = (-1)^{x+y} \frac{det(T' \setminus [y,x])}{det(T')}$$

Aby wyznaczyć krawędź dozwoloną wystarczy w macierzy T'^{-1} znaleźć element różny od zera, odpowiadający pewnej krawędzi. Następnie macierz odwrotną można zaktualizować bez potrzeby wyliczania jej od początku:

$$T'^{-1} = T'^{-1} - \frac{T'^{-1}[y,*] \cdot T'^{-1}[*,x]}{T'^{-1}[y,x]}$$

W ten sposób uzyskujemy prostą implementację algorytmu Muchy-Sankowskiego działającą w czasie $O(n^3)$. Wersja działająca w czasie $O(n^\omega)$ jest bardziej skomplikowana. Jej szczegółowy opis znajduje się w [23].

3 Opis wyników pracy

3.1 Skojarzenia w grafach kubicznych bez mostów

Punktem wyjścia dla algorytmu do wyznaczania doskonałego skojarzenia w grafach kubicznych bez mostów jest algorytm Biedl. Wąskim gardłem tego algorytmu jest weryfikacja czy graf jest dwuspójny, do której wykorzystywana jest dynamiczna struktura danych dla problemu dwuspójności grafu. Nowy algorytm (zaprezentowany w rozdziale 4.3 pracy) wykonuje test dwuspójności przy użyciu dwóch znacznie prostszych struktur danych — dynamicznej struktury danych dla problemu spójności grafu [16] oraz dynamicznych drzew Tarjan'a [28].

Dynamiczna struktura danych dla problemu spójności grafu jest wykorzystywana do dwóch rzeczy. Po pierwsze odpowiada ona na pytanie, czy dane dwa wierzchołki grafu są połączone ścieżką. Po drugie, utrzymuje ona drzewo rozpinające grafu, dla którego jest utworzona. Drzewo to służy do inicjalizacji i aktualizacji drzewa Tarjan'a. Ogólny zarys weryfikacji, czy wykonywana redukcja Frinke'a zachowuje dwuspójność grafu jest następująca:

- Jeśli któraś z par wierzchołków a, b, e lub f nie jest połączona ścieżką po wykonaniu redukcji, oznacza to, że należy wybrać drugi rodzaj redukcji.

- W przeciwnym przypadku, wykorzystując drzewo Tarjan'a (a dokładniej operację wyznaczania najbliższego wspólnego przodka dla danej pary wierzchołków), weryfikujemy, czy każdy najbliższy przodek wszystkich par wierzchołków a, b, e lub f leży na pewnym cyklu w zredukowanym grafie. Jak wykazano w pracy, jest to warunek konieczny i wystarczający do tego, aby analizowany graf kubiczny był dwuspójny.

W zależności od stosowanej wersji struktury danych dla problemu dynamicznej spójności uzyskuje się algorytm deterministyczny działający w czasie $O(n \log^2 n)$ (w przypadku struktury danych zaprezentowanej w [16]) lub algorytm randomizowany działający w czasie $O(n \log n \log\log^3 n)$ ([29]).

3.1.1 Użycie ubytkowych struktur danych

Podobnie jak w przypadku algorytmu Biedl, szybkość działania algorytmu z poprzedniego rozdziału jest ograniczona przez strukturę danych dla problemu spójności grafu. Redukcje Frinke'a nie tylko usuwają, ale również dodają krawędzie do grafu, więc użyta struktura danych musi wspierać obie operacje. W rozdziale 4.4 pracy prezentuję modyfikację algorytmu, która eliminuje potrzebę wspierania operacji dodawania krawędzi. Modyfikacja ta pozwala na stosowanie tzw. ubytkowych (ang. decremental) struktur danych.

W przypadku wielu klas grafów znane są ubytkowe wersje struktur danych znacznie szybsze od dynamicznych odpowiedników. Niestety, dla grafów rzadkich, a takimi są grafy kubiczne, nie jest znana szybsza struktura danych. Niewykluczone jednak, że w przyszłości zostanie ona skonstruowana.

3.2 Równoległy algorytm dla grafów dwudzielnych kubicznych

Kolejnym zaprezentowanym w pracy wynikiem (rozdział 5.1) jest równoległy algorytm znajdujący doskonałe skojarzenie w grafach dwudzielnych kubicznych. Działa on w czasie $O(\log^2 n)$ i wykorzystuje $O(n/\log n)$ procesorów. Algorytm ten jest tak samo szybki jak najszybszy znany do tej pory algorytm zaproponowany przez Shanan'a i Wigderson'a [27], ale jest znacznie prostszy i używa mniejszej liczby procesorów.

Algorytm wykorzystuje obserwację, że każdy dwudzielny (spójny) graf kubiczny jest dwuspójny (a zatem posiada również doskonałe skojarzenie). Zasada działania jest zbliżona do algorytmu Frinke'a. Główne różnice polegają na tym, że:

- algorytm równoległy wykonuje wiele redukcji jednocześnie względem zbioru niezależnych krawędzi,
- w fazie wycofywania redukcji wykonanych w jednym kroku cykl naprzemienny (wymagany przez przypadek z rysunku 2(d)) konstruowany jest tylko raz.

Do wyznaczenia zbioru niezależnych krawędzi wykorzystywany jest równoległy algorytm Shannon'a [26], który po zaadoptowaniu do naszych potrzeb znajduje zbiór niezależnych krawędzi liniowej wielkości w czasie $O(\log^* n)$ z użyciem $O(n)$ procesorów.

Pojedynczy krok wycofywania redukcji podzielony jest na dwie fazy. W pierwszej wycofywane są wszystkie redukcje, które nie mają obu krawędzi skojarzonych (przypadki (a), (b) i (c) z rysunku 2). W drugiej z grafu usuwane są wszystkie krawędzie skojarzone, co prowadzi do grafu regularnego stopnia 2. Następnie wyznaczane jest w nim doskonałe skojarzenie przy użyciu metody równoległego liczenia sum prefiksowych w czasie $O(\log n)$. Dzięki tej operacji wszystkie pozostałe redukcje mogą być wykonane, gdyż przypadek 2(d) został zamieniony w przypadek 2(a).

3.3 Równoległy algorytm dla grafów planarnych dwudzielnych kubicznych

Algorytm zaprezentowany w poprzednim punkcie wykonuje $O(\log n)$ faz, z których każda zabiera czas $O(\log n)$. Najwolniejszym elementem każdej fazy jest proces wyznaczania doskonałego skojarzenia rozłącznego z danym doskonałym skojarzeniem (metoda sum prefiksowych). Ograniczając się do analizy grafów planarnych można ten proces przyspieszyć.

W rozdziale 5.2 prezentuję algorytm, którego zasadniczą różnicą w stosunku do algorytmu z poprzedniego rozdziału jest wykonywanie redukcji względem zbioru niezależnych cykli grafu, a nie pojedynczych krawędzi. Redukując całe cykle na raz eliminuję potrzebę wyznaczania doskonałego skojarzenia rozłącznego z danym. Rozwiązanie takie prowadzi do algorytmu działającego w czasie $O(\log n \log^* n)$. W ten sposób uzyskałem najszybszy znany do tej pory algorytm do wyznaczania doskonałego skojarzenia w grafach planarnych dwudzielnych kubicznych.

3.4 Równoległy algorytm algebraiczny dla dowolnych grafów

Kolejna część pracy poświęcona jest równoległemu wyznaczaniu najliczniejszego skojarzenia w dowolnych grafach gęstych. Jako bazę wyjściową wykorzystany został algebraiczny algorytm Muchy i Sankowskiego [23]. W rozdziale 5.3 pracy zaprezentowane są kolejne wersje tego algorytmu. Rozdział 5.3.1 prezentuje wyznaczanie doskonałego skojarzenia w grafach dwudzielnych. Rozdział 5.3.2 uogólnia algorytm na przypadek dowolnych grafów. Rozdział 5.3.3 prezentuje sposób wyznaczania najliczniejszych, a nie tylko doskonałych skojarzeń.

Część teoretyczna pracy zakończona jest prezentacją teoretycznej implementacji algebraicznego algorytmu Muchy i Sankowskiego w wersji sekwencyjnej (rozdział 5.3.4) oraz w wersji równoległej dla architektury PRAM (rozdział 5.3.5). Druga część prezentacji algorytmu algebraicznego znajduje się w części praktycznej. Rozdział 6.2 przedstawia implementację algorytmu na architekturę dzisiejszych kart graficznych. W rozdziale tym omówione są również szczegóły architektury kart graficznych i problemy wydajnościowe, które trzeba mieć na względzie w celu uzyskania wydajnej implementacji algorytmu. Analizy wydajnościowe wykazują, że uzyskany algorytm dla pewnych klas grafów działa szybciej od klasycznych algorytmów sekwencyjnych szeroko stosowanych w praktyce. Dokładne wyniki przedstawione są w rozdziale 6.3.

3.5 Doskonałe skojarzenie poprzez redukcję do problemu SAT

Kolejny algorytm, zaprezentowany w rozdziale 6.1 rozprawy, konstruuje doskonałe skojarzenie poprzez konwersję grafu do formuły logicznej zawierającej $|E|$ zmiennych. Każdej krawędzi grafu jest przyporządkowana jedna zmienna. Formuła logiczna może zostać skonstruowana w sposób następujący — dla każdego wierzchołka $v \in V$ grafu z incydentnymi krawędziami $N_e(G, v) = \{e_1, e_2, \ldots, e_k\}$ generowane są następujące klauzule:

$$e_1 \vee e_2 \vee \cdots \vee e_k$$

$$\forall_{e_1, e_2 \in N_e(G,v), e_1 \neq e_2} \neg e_1 \vee \neg e_2$$

Wartościowanie spełniające formułę logiczną przypisuje wartość „prawda” wszystkim zmiennym, których odpowiadające krawędzie należą do skonstruowanego skojarzenia. Formuła nie jest spełnialna gdy graf nie posiada doskonałego skojarzenia.

Uzyskana w ten sposób formuła w przypadku grafów gęstych jest wielkości $\Theta(n^3)$, co czyni algorytm bezużytecznym (nawet, gdyby można było znaleźć wartościowanie spełnialne w czasie liniowym) w świetle istnienia algorytmu Micali'ego i Varizani'ego [21] działającego w czasie $o(n^3)$. Istnieje jednak możliwość wstępnego przetworzenia grafu wejściowego, która prowadzi do formuły liniowej wielkości, co szczegółowo opisano w rozdziale 6.1.

W ogólności problem znajdowania wartościowania spełniającego daną formułę należy do klasy NP. W naszym przypadku jednak znaczna część klauzul jest w postaci 2-CNF, co daje nadzieję na stosunkowo krótki czas rozwiązywania problemu w praktyce. Rozwiązywanie problemu SAT może być szczególnie szybkie w przypadku istnienia wielu rozwiązań. Ograniczając się do grafów kubicznych dwuspójnych nadzieję taką daje hipoteza Lovasz'a i Plummer'a:

Hipoteza 1 (Lovasz-Plummer)**.** *Niech G będzie dowolnym grafem kubicznym bez mostów o n wierzchołkach. Istnieje stała c taka, że G posiada co najmniej e^{cn} doskonałych skojarzeń.*

Jak pokazują eksperymenty zaprezentowane w rozdziale 6.3.3, algorytm do wyznaczania doskonałych skojarzeń poprzez redukcję do problemu SAT działa w czasie liniowym dla pewnych klas grafów.

3.6 Analizy wydajnościowe algorytmów

W ramach podsumowania uzyskanych wyników z praktycznego punktu widzenia dokonałem porównania wydajności zaproponowanych algorytmów (rozdziały 6.3 i 6.4). Analizie zostały poddane następujące algorytmy:

- implementacja algorytmu Edmondsa pochodząca z biblioteki Boost [1]. Pod uwagę zostały wzięte cztery wersje tego algorytmu — wersja bez zastosowania heurystyk wyznaczających początkowe skojarzenie oraz algorytmy wykorzystujące trzy różne metody do wyznaczania początkowego skojarzenia,
- algorytm do wyznaczania doskonałego skojarzenia w dowolnych grafach metodą SAT;
- równoległy algorytm macierzowy zrealizowany na karcie graficznej do wyznaczania najliczniejszego skojarzenia;
- algorytm Biedl do wyznaczania doskonałego skojarzenia w grafach kubicznych dwuspójnych;
- nowy algorytm do wyznaczania doskonałego skojarzenia w grafach kubicznych dwuspójnych.

Działanie algorytmów zostało zbadane na 10-ciu grupach grafów (w sumie 5950 grafów zawierających ponad 100 milionów wierzchołków i ponad 800 milionów krawędzi) pochodzących z różnych źródeł i zawierających m.in. grafy planarne, regularne i dwudzielne.

Analiza uzyskanych wyników wykazała między innymi, że w przypadku grafów gęstych najszybszym algorytmem jest równoległy algorytm macierzowy. Jest to dość obiecujący rezultat, który ukazuje sens prowadzenia dalszych badań w zakresie równoległych algorytmów grafowych.

Algorytm bazujący na redukcji do problemu SAT w przypadku pewnych grafów działa bardzo długo. Okazuje się jednak, że ograniczając się do grafów dwuspójnych kubicznych (które, zgodnie z hipotezą posiadają wykładniczo wiele skojarzeń), w przypadku średnim algorytm zachowuje się bardzo dobrze. Tylko jeden spośród badanych algorytmów jest w praktyce szybszy — algorytm Edmondsa z udoskonaloną przeze mnie metodą konstruowania początkowego skojarzenia.

Algorytm z rozdziału 3.1 do wyznaczania doskonałego skojarzenia w grafach dwuspójnych kubicznych dla większości testowanych grafów przegrywa z algorytmem Edmondsa i algorytmem SAT, które działają średnio w czasie liniowym. W przypadkach

pesymistycznych pozostaje on jednak daleko przed konkurencją, co jest potwierdzeniem teoretycznych złożoności badanych algorytmów.

Literatura

[1] Boost, C++ Libraries. *http://www.boost.org/*.

[2] M. G. Andrews, M. J. Atallah, D. Z. Chen, and D. T. Lee. Parallel algorithms for maximum matching in interval graphs. In *Proceedings of the 9th International Symposium on Parallel Processing*, IPPS '95, pages 84–92, Washington, DC, USA, 1995. IEEE Computer Society.

[3] T. Biedl. Linear reductions of maximum matching. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, SODA '01, pages 825–826, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.

[4] T. C. Biedl, P. Bose, E. D. Demaine, and A. Lubiw. Efficient algorithms for Petersen's matching theorem. In *Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '99, pages 130–139, Philadelphia, PA, USA, 1999. Society for Industrial and Applied Mathematics.

[5] A. B. Borodin, J. Von zur Gathen, and J. E. Hopcroft. Fast Parallel Matrix and GCD Computations. Technical report, Ithaca, NY, USA, 1982.

[6] P. Chaudhuri. Finding maximum matching for bipartite graphs in parallel. 1994.

[7] M. K. E. Dahlhaus and A. Lingas. A Parallel Algorithm for Maximum Matching in Planar Graphs. *TR-89-018*.

[8] J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[9] J. Egerváry. Matrix kombinatorius tulajdonságairól. *Matematikai és Fizikai Lapok*, 38:16–28, 1931.

[10] T. Feder and R. Motwani. Clique partitions, graph compression and speeding-up algorithms. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, STOC '91, pages 123–133, New York, NY, USA, 1991. ACM.

[11] C. Fremuth-Paeger and D. Jungnickel. Balanced network flows. VIII. A revised theory of phase-ordered algorithms and the $O(\log(n^2/m)/\log n)$ bound for the nonbipartite cardinality matching problem. *Networks*, 41(3):137–142, 2003.

[12] O. Frink. A Proof of Petersen's Theorem. *The Annals of Mathematics*, 27:491–493, 1926.

[13] H. Gabow. An efficient implementation of Edmonds' algorithm for maximum matching in graphs. *Journal of the ACM*, 23(2):221–234, 1976.

[14] K. M. H. Alt, N. Blum and M. Paul. Computing a Maximum Cardinality Matching in a Bipartite Graph in Time $O(n^{1.5}\sqrt{m/\log n})$. *Information Processing Letters*, 37:237–240, 1991.

[15] N. J. A. Harvey. Algebraic Algorithms for Matching and Matroid Problems. *SIAM J. Comput.*, 39:679–702, July 2009.

[16] J. Holm, K. de Lichtenberg, and M. Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 79–89, New York, NY, USA, 1998. ACM.

[17] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[18] R. M. Karp, E. Upfal, and A. Wigderson. Constructing a perfect matching is in random NC. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*, STOC '85, pages 22–32, New York, NY, USA, 1985. ACM.

[19] D. König. Graphok és matrixok. *Matematikai és Fizikai Lapok*, 38:116–119, 1931.

[20] P. T. M. Mucha. Finding maximum matchings via Gaussian elimination, Warsaw University, Faculty of Mathematics, Informatics and Mechanics.

[21] S. Micali and V. V. Vazirani. An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs. In *Proceedings of the 21st Annual IEEE Symposium on Foundations of Computer Science*, pages 17–27, 1980.

[22] M. Mucha and P. Sankowski. Maximum Matchings in Planar Graphs via Gaussian Elimination. In *Proceedings of the 12th Annual European Symposium on Algorithms*, pages 532–543, 2004.

[23] M. Mucha and P. Sankowski. Maximum Matchings via Gaussian Elimination. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, pages 248–255, Washington, DC, USA, 2004. IEEE Computer Society.

[24] J. Petersen. Die Theorie der regulären Graphs. *Acta Mathematica*, 15:193–220, 1891.

[25] A. Schrijver. Bipartite Edge Coloring in $O(\Delta m)$ Time. *SIAM Journal on Computing*, 28:841–846, 1999.

[26] G. Shannon. Parallel Independent Set Algorithms for Sparse Graphs. *CSD-TR-634*, 1986.

[27] R. Sharan and A. Wigderson. A new NC algorithm for perfect matching in bipartite cubic graphs. In *Proceedings of ISTCS 96*, pages 56–65, 1996.

[28] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *J. Comput. Syst. Sci.*, 26:362–391, June 1983.

[29] M. Thorup. Near-optimal fully-dynamic graph connectivity. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, STOC '00, pages 343–350, New York, NY, USA, 2000. ACM.

[30] W. T. Tutte. The factors of graphs. *Canadian Journal of Mathematics 4*, pages 314–328, 1952.