# 1. Przegląd literatury

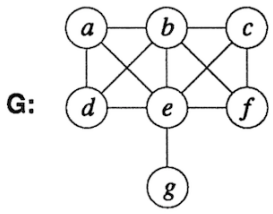
## 1.1. Istniejące metody partycjonowania grafów

1. Rozbudowany podział metod został zaproponowany przez autorów artykułu - [16] oraz [24] Zgodnie z ich analizą metody dzielimy na:

- spektralne,
- rekursywne,
- geometryczne,
- wielopoziomowe.

### 1.1.1. Metody spektralne

Partycjonowanie spektralne daje dobre rezultaty i jest metodą w miarę często używaną [26, 27, 18]. Podzbiór wierzchołków $S \subset V$ grafu G nazywamy separatorem jeśli dwa wierzchołki w tym samym komponencie grafu G są w dwóch różnych komponentach w grafie $G \setminus S$.



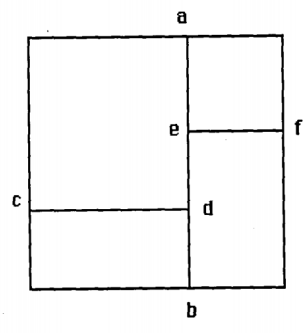

Rysunek 1: Separator wierzchołków d oraz c to $\{b, e\}$. Źródło: [22].

Metoda [26] będąca algorytmem spektralnym pokazuje algebraiczne podejście do obliczania separatorów wierzchołków. Artykuł [27] opisuje algorytm spectral nested dissection (SND). Algorytmy te za pomocą znajomości spektralnych właściwości macierzy Laplaciana obliczają separatory wierzchołków w grafie, które tworzą partycjonowanie grafu. Artykuł [18] mówi o nowatorskim rozwinięciu metody spektralnej pod kątem umożliwienia podziału obliczeń na cztery bądź osiem części na każdym etapie rekursywnej dekompozycji. Są to jednak wszystko metody kosztowne z racji na obliczanie wektorowa własnego odpowiadającego drugiej najmniejszej wartości własnej (Fielder wektor). Istnieją udane próby ulepszenia czasu wykonania tych metod, które polegają na liczeniu Fielder wektora poprzez algorytm wielopoziomowy - MSB [1]. Jednak nawet te metody wciąż charakteryzują się wysoką złożonością.

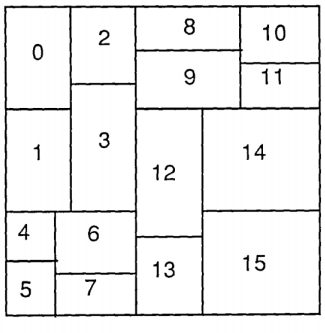
### 1.1.2. Metody rekursywne

Są to metody często prostsze w implementacji, jednak nie sprawdzają się tak dobrze w kontekście bardziej skomplikowanych problemów głównie ze względu na to, że mają zachłanną naturę. Metoda [2] zakłada, że dzielimy siatkę na liczbę obszarów, która jest

równa potędze liczby dwa. Ta metoda potrafi także dzielić siatkę wedle możliwości obliczeniowych poszczególnych rdzeni procesora. Czasami metody rekursywne są implementowane jako faza metod bisekcji spektralnej [26]. Przykłady działania partycjonowania rekursywnego przedstawione są na rysunkach 2 oraz 3. Specyfika tych metod polegająca na dzieleniu grafu na coraz mniejsze części sprawia, że nie jesteśmy w stanie uwzględniać części niepodzielnych, lub rozwiązanie tego problemu byłoby skomplikowane. Przykładem jest sytuacja kiedy dzielimy siatkę na 4 części. Można sobie wyobrazić sytuację, kiedy obszar niepodzielny zajmuje 50% całej siatki. Po pierwszej turze rekursywnego algorytmu mamy dwie partycje, każda zajmująca 50% powierzchni. Chcielibyśmy podzielić każdą z nich na dwie części, natomiast nie jesteśmy w stanie tego zrobić ponieważ jedna z nich jest w całości obszarem niepodzielnym. Wykorzystanie takich algorytmów w kontekście niepodzielnych obszarów nie zdaje więc egzaminu.




Rysunek 2: Przedstawia partycjonowanie rekursywne na głębokości wynoszącej 2. Linia partycjonowania a-b, która została stworzona przez partycjonowanie poziomu 1 jest podzielona na 3 segmenty przez dwie linie partycjonowania poziomu drugiego: c-d, e-f. Źródło: [2]




Rysunek 3: Metoda rekursywna - binarna dekompozycja dla 16 procesorów. Najpierw tworzone jest wertykalne cięcie, które gwarantuje, że prawy i lewy obszar zawiera połowę pracy do wykonania (lub takie, które jest jak najbliżej takiego podziału). Jeśli dostępne są 4 procesory to każdy z dwóch segmentów partycjonowany jest horyzontalną linią, która spełnia te same założenia jak ta dla pierwszego wertykalnego cięcia. Procedura jest kontynuowana na zmianę wykorzystując wertykalne i horyzontalne cięcie aż do otrzymania podziału na oczekiwaną liczbę obszarów. Źródło: [2]

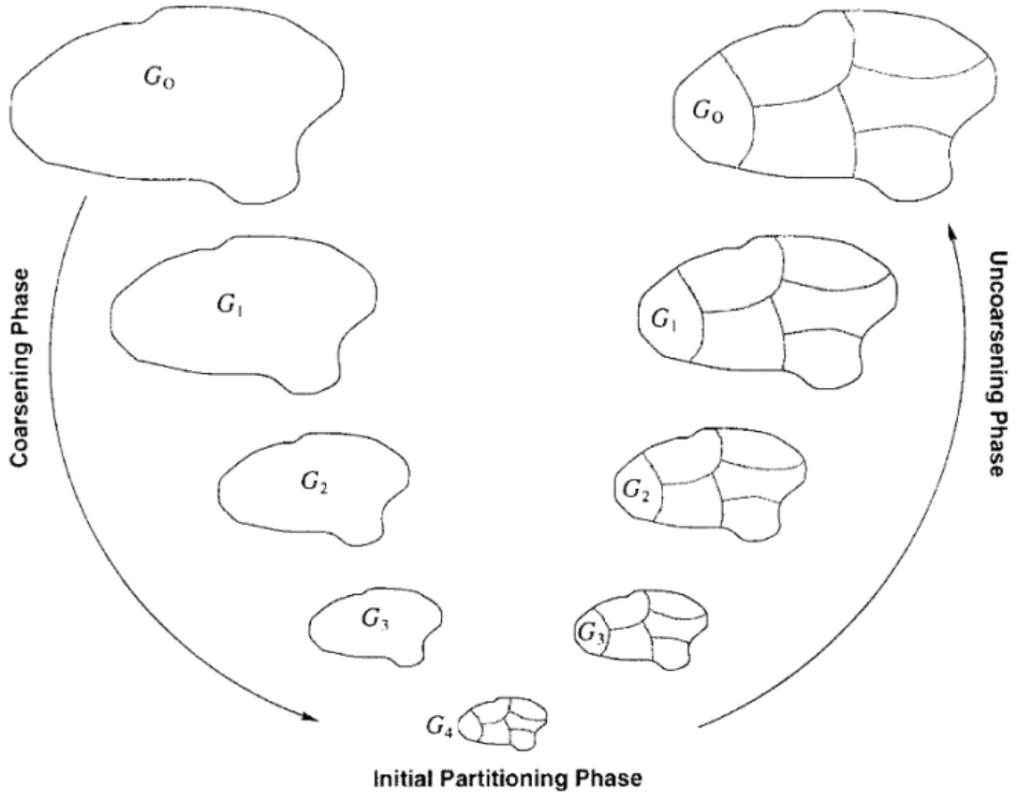
### 1.1.3. Metody geometryczne

Inną klasą metod są metody geometryczne [20, 29, 21, 22, 25]. Używają danych geometrycznych o grafie w celu znalezienia dobrego partycjonowania. Ich cechą charakterystyczną jest szybki czas wykonania, natomiast gorsze rezultaty podziału niż metody spektralne. Najlepsze wyniki spośród wyżej wymienionych metod prezentują [21, 22]. W artykule [21] zaproponowane klasę grafów zwaną k-overlap graphs. Dla grafów k-overlap osadzonych w d wymiarach [32] udowadniają istnieje separatora wielkości $O(k^{1/d}D^{d-1/d})$. Oznacza to, że dla grafu o N wierzchołkach istnieje podzbiór wierzchołków o wcześniej wymienionej wielkości, który po usunięciu rozłącza graf na dwie podobnej wielkości części. Wyniki w tym artykule unifikują kilka wcześniejszych wyników dla obliczania separatorów. Ponadto autorzy proponują rozwiązanie, które oblicza separatory w czasie liniowym. W artykule [22] opisano efektywną metodę partycjonowania siatek niestrukturalnych, która występują w metodach elementów skończonych i różnic skończonych. Podejście to wykorzystuję strukturę geometryczną danej siatki i znajduje dobre partycjonowanie w czasie $O(n)$. Można je aplikować do siatek w dwóch i trzech wymiarach. Ma zastosowanie w wydajnych algorytmach sekwencyjnych i równoległych do rozwiązywania rozbudowanych problemów w obliczeniach naukowych. Charakterystyką metod geometrycznych jest to, że z powodu losowej natury wymagane jest wielokrotne użycie algorytmu (od 5 do 50 razy) aby uzyskać wynik porównywalny z metodami spektralnymi. Wielokrotnie wywołanie zwiększa czas otrzymywania rezultatu, natomiast jest on wciąż niższy od metod spektralnych. Metody geometryczne są aplikowalne tylko w przypadku kiedy dostępne są współrzędne wszystkich wierzchołków w grafie. Dla wielu dziedzin problemów (programowanie liniowe, VLSI), nie otrzymujemy współrzędnych wraz z grafem. Istnieją algorytmy, które są w stanie obliczyć współrzędne dla wierzchołków grafu [4] wykorzystując metody spektralne ale są bardzo kosztowne i dominują czas potrzebny na samo partycjonowanie grafu. Implementacje tego typu metod nie znalazły się wśród algorytmów stat-of-the-art dla problemu partycjonowania grafów, więc nie były brane pod uwagę w kontekście niniejszej pracy.

Rysunek 4: Rekursywne partycjonowanie mapy USA - pierwszy obrazek od góry - za pomocą algorytmu artykułu [22]. Dwa następne obrazki prezentują rezultat. Dane geometryczne używane są do obliczenia separatorów.

### 1.1.4. Metody wielopoziomowe

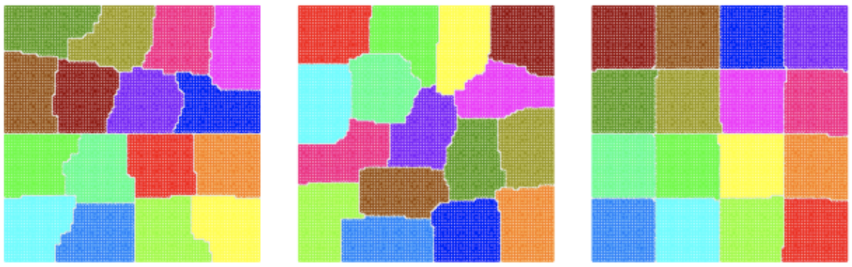
Istnieje wiele implementacji metod wielopoziomowych: [16, 31, 3, 5, 10, 9, 11, 8, 19]. Cechą charakterystyczną tego podejścia jest redukcja wielkości grafu poprzez łączenie wierzchołków i krawędzi, następnie dzielenie zmniejszonego grafu na partycje, ostatnią fazą jest przywrócenie początkowego grafu zachowując podział. Często graf zmniejszany jest aż liczba wierzchołków nie osiągnie liczby partycji, którą chcemy otrzymać [24], a fazie przywracania grafu do początkowej wielkości towarzyszy algorytm, którego celem jest ulepszanie podziału [6, 7]. Algorytm ten, bazując na zmniejszonym grafie, niesie za sobą niższy koszt obliczeniowy. Jego działanie polega na zmniejszaniu długości granic pomiędzy partycjami z jednoczesnym zachowaniem ich wielkości. Na tym etapie może także zostać dodana faza balansowania, która stopniowo zmniejsza różnice w wielkości pól pomiędzy obszarami. Metody te zostały stworzone z myślą o zmniejszeniu czasu patrycjonowania kosztem jego jakości. Obecnie dają jednak bardzo dobre rezultaty również w kwestii jakości podziału. Późniejsze prace w dziedzinie tych algorytmów pokazały, że dają one lepsze rezultaty niż metody spektralne [16]. Biblioteki jak Party [24], Metis [16], Jostle [31], Chaco [11], dające state-of-the-art wyniki w kwestii jakości partycjonowania najczęściej bazują na schemacie wielopoziomowym [11].




Rysunek 5: Wielopoziomowe partycjonowanie grafu przedstawiające fazę zmniejszania grafu, następnie przypisanie partycji na zmniejszonym grafie, na końcu przywrócenie grafu do początkowej wielkości. Źródło: [14].

## 1.2. Porównanie istniejących metod wielopoziomowych

Wszystkie wyżej wymienione metody nie biorą pod uwagę problemu obszarów niepodzielnych oraz obszarów wyłączonych z obliczeń. W związku z tym celem niniejszej pracy stało się znalezienie metody dającej możliwie najlepsze rezultaty w zakresie partycjonowania grafów oraz dostosowanie jej do wyżej wymienionych rozszerzeń problemu partycjonowania.
Metody wielopoziomowe były najlepszym wyborem, z racji na to, że gwarantowały najlepsze wyniki partycjonowania. Przykładów metod wielopoziomowych było bardzo dużo [16, 31, 3, 5, 10, 9, 11, 8, 19], jednak skupiłem się głównie na tych, które dawały wyniki state-of-the-art. Były to biblioteki: Party [24], Metis [16], Jostle [31], Chaco [11].



Rysunek 6: Partycjonowanie siatki 100x100 na 16 obszarów. Od lewej - pmetis [16] uzyskuje edge-cut wynoszący 688, następnie Jostle [31] z wynikiem 695 oraz Party [24] z wynikiem 615. Źródło: [24].

Do zmniejszenia grafu stosowane są różne warianty matching algorithm (algorytm do budowania skojarzeń w grafie). Porównanie większości z nich można znaleźć w [13]. Przykładowo biblioteka Jones oraz Bui [3] używam random weighted matching, natomiast Party [24] stosuje LAM matching [28]. Ze względu na wymagania co do złożoności obliczeniowej wszystkie metody używają heurystyk. Wszystkie z tych metod zmniejszają graf, jednak tylko Jostle i Party zmniejsza graf aż do otrzymania liczby wierzchołków równej liczbie partycji, na które chcemy podzielić wejściowy graf. Dzięki temu metoda partycjonowania, działająca na najmniejszym możliwym grafie, jest dużo prostsza niż w pozostałych metodach. Kiedy graf jest zmniejszony, wierzchołki są przyporządkowywane do partycji a informacja na temat partycji jest propagowana do wierzchołków na wyższych poziomach zgodnie z partycją ich reprezentantów na najniższym poziomie. Ten proces prowadzi do partycjonowania początkowego grafu. Faza zmniejszania grafu jest ponadto stosunkowo łatwa do zrównoleglenia [15]. Fazą, która nie podlega zrównolegleniu jest faza ulepszania istniejącego podziału. Najczęściej bazuje ona na metodzie Fiduccia-Mattheysesa [7], która jest zoptymalizowaną pod kątem czasu działania heurystyką Kerninghan-Lin (KL) [17]. Artykuł [17] podejmuje problem partycjonowania wierzchołków grafu, którego krawędzie mają przyporządkowane wagi - koszt. Jego celem jest ustanowienie podziału na obszary o wybranej wielkości wraz ze zminimalizowaniem sumy kosztów na wszystkich granicach. Artykuł [7] przedstawia heurystykę do poprawiania partycjonowania sieci. Algorytm ten przesuwa pojedyncze komórki pomiędzy blokami wraz z utrzymaniem oczekiwanego rozmiaru bloków. W przeciwieństwie do Metis i Jostle, faza ulepszenia partycjonowania używana przez biblioteke Party bazuje na metodzie Helpful-Sets (HS). Heurystyka Helpful-Set

wywodzi się z obserwacji teoretycznych wykorzystywanych do znajdowania górnych granic szerokości bisekcji grafów regularnych [12, 23]. Szerokość bisekcji to minimalna liczba krawędzi, która musi zostać usunięta w celu podzielenia grafu na dwie równej wielkości części, lub różniące się wielkością o maksymalnie jeden wierzchołek. Party otrzymuje bardzo dobre wyniki stosując tę metodę [30], często uzyskując mniejszą długość granic pomiędzy obszarami niż Metis czy Jostle, jednocześnie jest tylko trochę bardziej kosztowna obliczeniowo. Heurystyka Helpful-Sets jest metodą bazującą na wyszukiwaniu lokalnym. Zaczynając od początkowej bisekcji $\pi$ dąży do zminimalizowania długości granicy za pomocą lokalnie wprowadzanych zmian. Najważniejsza różnica względem metody KL polega na tym że KL przemieszcza tylko pojedyncze wierzchołki, natomiast HS zbiory wierzchołków. Zarówno Party, Metis jak i Jostle pozwalają na małe nierówności w kwestii wielkości obszarów co przekłada się na mniejsze długości granic, szczególnie na głębszych poziomach, kiedy przemieszczane są wierzchołki o dużych wagach - ciężko wtedy o otrzymanie idealnie równych obszarów.
Biblioteka Party [24] okazała się dawać najlepsze rezultaty w porównaniu do innych bibliotek dających wyniki state-of-the-art. Szczególną własnością Party była znacznie niższa długość granic w porównaniu do pozostałych bibliotek [Rysunek 6], co było bardzo ważne dla mojej pracy. Dlatego to właśnie metodę z biblioteki Party zdecydowałem się wybrać jako podstawę rozwiązania prezentowanego w niniejszej pracy.

# Materiały źródłowe


[1] S. Barnard and H. Simon. A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problems. pages 711–718, 01 1993.

[2] M. Berger and S. Bokhari. A partitioning strategy for nonuniform problems on multiprocessors. *IEEE Transactions on Computers*, C-36, 06 1987.

[3] T. N. Bui and C. Jones. A heuristic for reducing fill-in in sparse matrix factorization. In *PPSC*, 1993.

[4] T. F. Chan, J. R. Gilbert, and S.-H. Teng. Geometric spectral partitioning. Technical report, 1995.

[5] C.-K. Cheng and Y.-C. Wei. An improved two-way partitioning algorithm with stable performance (vlsi). *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 10(12):1502–1511, 1991.

[6] R. Diekmann and B. Monien. Using helpful sets to improve graph bisections. 08 1994.

[7] C. M. Fiduccia and R. M. Mattheyses. A linear-time heuristic for improving network partitions. In *Proceedings of the 19th Design Automation Conference*, DAC '82, page 175–181. IEEE Press, 1982.

[8] J. Garbers, H. Promel, and A. Steger. Finding clusters in vlsi circuits. In *1990 IEEE International Conference on Computer-Aided Design. Digest of Technical Papers*, pages 520–523, 1990.

[9] Hagen and Kahng. A new approach to effective circuit clustering. In *1992 IEEE/ACM International Conference on Computer-Aided Design*, pages 422–427, 1992.

[10] L. Hagen and A. Kahng. Fast spectral methods for ratio cut partitioning and clustering. In *1991 IEEE International Conference on Computer-Aided Design Digest of Technical Papers*, pages 10–13, 1991.

[11] B. Hendrickson and R. Leland. A multi-level algorithm for partitioning graphs. pages 28– 28, 02 1995.

[12] J. Hromkovič and B. Monien. The bisection problem for graphs of degree 4 (configuring transputer systems). In A. Tarlecki, editor, *Mathematical Foundations of Computer Science 1991*, pages 211–220, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.

[13] G. Karypis and V. Kumar. Analysis of multilevel graph partitioning. pages 29– 29, 02 1995.

[14] G. Karypis and V. Kumar. Multilevelk-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing*, 48(1):96–129, 1998.

[15] G. Karypis and V. Kumar. A parallel algorithm for multilevel graph partitioning and sparse matrix ordering. *Journal of Parallel and Distributed Computing*, 48(1):71–95, 1998.

[16] G. Karypis and V. Kumar. Kumar, v.: A fast and high quality multilevel scheme for partitioning irregular graphs. siam journal on scientific computing 20(1), 359-392. *Siam Journal on Scientific Computing*, 20, 01 1999.

[17] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *The Bell System Technical Journal*, 49(2):291–307, 1970.

[18] R. Leland and B. Hendrickson. An improved spectral graph partitioning algorithm for mapping parallel computations. 16, 09 1992.

[19] N. Mansour, R. Ponnusamy, A. Choudhary, and G. C. Fox. Graph contraction for physical optimization methods: A quality-cost tradeoff for mapping data on parallel computers. In *Proceedings of the 7th International Conference on Supercomputing*, ICS '93, page 1–10, New York, NY, USA, 1993. Association for Computing Machinery.

[20] G. Miller, S. Teng, and W. Thurston. A cartesian parallel nested dissection algorithm. 1994.

[21] G. Miller, S.-H. Teng, and S. Vavasis. A unified geometric approach to graph separators. In *[1991] Proceedings 32nd Annual Symposium of Foundations of Computer Science*, pages 538–547, 1991.

[22] G. L. Miller, S. Teng, W. Thurston, and S. A. Vavasis. Automatic mesh partitioning. In A. George, J. Gilbert, and J. Liu, editors, *Graphs Theory and Sparse Matrix Computation*, The IMA Volumes in Mathematics and its Application, pages 57–84. Springer-Verlag, 1993. Vol 56.

[23] B. Monien and R. Preis. Upper bounds on the bisection width of 3- and 4-regular graphs. *Journal of Discrete Algorithms*, 4(3):475–498, 2006. Special issue in honour of Giorgio Ausiello.

[24] B. Monien and S. Schamberger. Graph partitioning with the party library: helpful-sets in practice. In *16th Symposium on Computer Architecture and High Performance Computing*, pages 198–205, 2004.

[25] B. Nour-Omid, A. Raefsky, and G. Lyzenga. Solving finite element equations on concurrent computers. 1987.

[26] A. Pothen, H. D. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. Matrix Anal. Appl.*, 11(3):430–452, May 1990.

[27] A. Pothen, H. D. Simon, L. Wang, and S. T. Barnard. Towards a fast implementation of spectral nested dissection. In *Proceedings of the 1992 ACM/IEEE Conference on Supercomputing*, Supercomputing '92, page 42–51, Washington, DC, USA, 1992. IEEE Computer Society Press.

[28] R. Preis. Linear time 1/2 -approximation algorithm for maximum weighted matching in general graphs. In *Proceedings of the 16th Annual Conference on Theoretical Aspects of Computer Science*, STACS'99, page 259–269, Berlin, Heidelberg, 1999. Springer-Verlag.

[29] P. Raghavan. Line and plane separators. Technical report, LAPACK WORKING NOTE 63 (UT CS-93-202), 1993.

[30] S. Schamberger. Improvements to the helpful-set algorithm and a new evaluation scheme for graph-partitioners. In V. Kumar, M. L. Gavrilova, C. J. K. Tan, and P. L'Ecuyer, editors, *Computational Science and Its Applications — ICCSA 2003*, pages 49–53, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[31] C. Walshaw and M. Cross. Mesh partitioning: A multilevel balancing and refinement algorithm. *SIAM Journal on Scientific Computing*, 22, 07 2004.

[32] Wikipedia. Graph embedding — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Graph%20embedding&oldid=1019397582`, 2021. [Online; accessed 28-June-2021].