ТЕХНИЧЕСКИ УНИВЕРСИТЕТ-ГАБРОВО
Факултет „Електротехника и електроника”

маг. инж. Матьо Стефанов Динев

ИЗСЛЕДВАНЕ НА АЛГОРИТМИ ЗА ДЕКОМПОЗИЦИЯ

А В Т О Р Е Ф Е Р А Т

на дисертация
за придобиване на образователна и научна степен „доктор”

Област на висше образование: 5. „Технически науки“

Професионално направление: 5.3. „Комуникационна и компютърна техника“

Докторска програма: „Автоматизация на инженерния труд и системи за автоматизирано проектиране“.

Габрово, 2023 г.

ТЕХНИЧЕСКИ УНИВЕРСИТЕТ-ГАБРОВО
Факултет „Електротехника и електроника”

маг. инж. Матьо Стефанов Динев

ИЗСЛЕДВАНЕ НА АЛГОРИТМИ ЗА ДЕКОМПОЗИЦИЯ

А В Т О Р Е Ф Е Р А Т

на дисертация

за придобиване на образователна и научна степен „доктор”

Област на висше образование: 5. „Технически науки“

Професионално направление: 5.3. „Комуникационна и компютърна техника“

Докторска програма: „Автоматизация на инженерния труд и системи за автоматизирано проектиране“.

Научен ръководител: доц. д-р. инж. Валентина Стоянова Кукенска

Рецензенти: проф. дтн. Стойчо Димитров Стойчев

проф. дмн. Стоян Недков Капралов

Габрово, 2023 г.

Дисертационният труд е обсъден и насочен за официална защита на заседание на Разширен катедрен съвет на катедра „Компютърни системи и технологии” към факултет „Електротехника и електроника” на Технически университет – Габрово, проведен на 06.02.2023г.

Дисертационният труд съдържа 136 страници. Научното съдържание е представено в увод, 5 глави и заключение и включва 98 фигури и 26 таблици и 48 формули. Цитирани са 118 литературни източника. Номерацията на фигурите, таблиците и формулите в автореферата е в съответствие с тази в дисертацията.

Разработката и изследванията по дисертационния труд са извършени в катедра „Компютърни системи и технологии” към факултет „Електротехника и електроника” на Технически университет – Габрово.

Официалната защита на дисертационния труд ще се състои на ..... г. от ..... ч. в зала ..... на Технически университет – Габрово.

Благодарности:

Благодаря на катедра „Компютърни системи и технологии“ на Технически Университет – Габрово за предоставените ресурси и възможности, които подпомогнаха моето изследване.

Благодаря на рецензентите проф. дтн. Стойчо Стойчев и проф. дмн. Стоян Капралов за направените забележки и препоръки, в резултат на които дисертационният труд придоби завършения си вид.

Изказвам своята благодарност на моя научен ръководител доц. д-р инж. Валентина Кукенска за ценната подкрепа, надзор, насоки и редакторска помощ, които се оказаха решаващи за завършването на настоящия дисертационен труд.

Автор: Матьо Стефанов Динев

Заглавие: ИЗСЛЕДВАНЕ НА АЛГОРИТМИ ЗА ДЕКОМПОЗИЦИЯ

А. ОБЩА ХАРАКТЕРИСТИКА НА ДИСЕРТАЦИОННИЯ ТРУД

АКТУАЛНОСТ НА ПРОБЛЕМА

Изборът на подходящ алгоритъм за решаването на дадена задача е актуален проблем, който е поставян и решаван в редица изследвания през последните 50 години.

Съществуват алгоритми, които все още не са достатъчно проучени. Такива са алгоритмите за декомпозиция, което обуславя **необходимостта да бъдат изследвани**, както и да се **направи оценка на тяхната сложност**.

ЦЕЛ И ЗАДАЧИ НА ДИСЕРТАЦИОННИЯ ТРУД

Целта на настоящия дисертационен труд е да се **изследват алгоритми за декомпозиция, да се направят оценки за тяхната сложност и да се приложат при решаване на задачи** от областта на автоматизираното проектиране на схеми и системи.

За постигане на поставената цел са формулирани следните **задачи**:

1) Да се направи класификация на алгоритмите за декомпозиция;
2) Да се направи аналитична оценка за сложността на изследваните алгоритми;
3) Да се разработят програми приложения, реализиращи алгоритмите за декомпозиция;
4) Да се изследват реализираните алгоритми;
5) Да се приложат алгоритмите за декомпозиция при решаване на задачи от функционално и конструктивно проектиране на технически схеми и системи.

ОБЕКТ И ПРЕДМЕТ НА ИЗСЛЕДВАНЕ

Обект на изследване в настоящата работа са **декомпозиционните алгоритми**, а **предмет** – **графовите модели** на различни обекти.

МЕТОДИ НА ИЗСЛЕДВАНЕ

Методите за изследване са обособени в отделните глави, като се използват аналитични, математически и практически похвати. Обхващат се зависимостите между получените подграфи след разделянето на графови модели и се правят изводи за използването на алгоритмите за декомпозиция.

НАУЧНА НОВОСТ

Предложена е класификация на алгоритмите за декомпозиция.

По правилата за оценяване на алгоритмите и чрез математически методи, приложени върху процедурните стъпки на изследваните алгоритми, са

изведена изрази за оценка на сложност за всеки един разглежданите алгоритми.

ПРИЛОЖИМОСТ

Изследваните алгоритми и разработените програми намират приложение при решаването на задачи, за които е необходимо да се използва принципът на декомпозиция. Те биха могли да намерят място при: построяването на модели, изследването на обекти с големи размери, проектирането на схеми, системи и устройства и др.

Получените резултати от изследването могат да се използват при избора на подходящ алгоритъм за решаване на задачи, както и при бъдещи изследвания в същата или подобна област.

АПРОБАЦИЯ НА ДИСЕРТАЦИОННИЯ ТРУД

Дисертационната работа е докладвана и обсъждана на катедрен съвет и на разширен катедрен съвет на катедра „Компютърни системи и технологии" при Технически университет – Габрово.

Основните резултати от дисертационния труд са публикувани и докладвани в международни научни конференции:

- XII International Conference Strategy of Quality in Industry and Education, May 30 –June 2, 2016, Varna.
- XXV Международна научна конференция „Мениджмънт и качество", Ямбол, 11-12 май 2016.
- Международна научна конференция Унитех'16 – Габрово, 18-19 ноември 2016.
- Международна научна конференция „Техника, Технологии, Образование" – ICTTE 2017, Ямбол, 19-20 октомври 2017.
- V – научна конференция с международно участие „Компютърни науки и технологии", Варна, 28-29 септември 2018.
- Международна научна конференция Унитех'18 – Габрово, 16-17 ноември 2018.
- Международна научна конференция Унитех'19 – Габрово, 15-16 ноември 2019.
- Международна научна конференция Унитех'20 – Габрово, 20-21 ноември 2020.
- Международна научна конференция Унитех'22 – Габрово,19-20 ноември 2022.

СТРУКТУРА И ОБЕМ НА ДИСЕРТАЦИОННИЯ ТРУД

Дисертационният труд съдържа увод, пет глави, заключение и списък на използваната литература с общ обем от 136 страници. Включени са 98 фигури под формата на схеми, графики и диаграми, 26 таблици и 48 формули.

Номерацията на фигурите, таблиците, формулите и цитираната литература в автореферата съответства на номерацията в дисертационния труд.

Б. СЪДЪРЖАНИЕ НА ДИСЕРТАЦИОННИЯ ТРУД

Глава 1. АНАЛИЗ НА СЪСТОЯНИЕТО НА ПРОБЛЕМА

В тази глава са описани основните характеристики и свойства на алгоритмите, както и критериите за оценяването им. Представени са алгоритмите за декомпозиция, които са обект на изследване. Предложена е класификация на декомпозиционните алгоритми. Формулирани са целта и задачите на дисертационния труд.

Оценката на алгоритмите може да се направи по различни критерии. Те са:

- Време за изпълнение;
- Необходима памет;
- Дефиниционна област;
- Точност.

От тях най-често се използват времето за изпълнение и необходимата памет. Акцент в настоящия дисертационен труд е времето за изпълнение на представените алгоритми.

Декомпозицията е един от основните принципи на системния подход. Тя се състои в разделянето на системата на подсистеми, докато е необходимо или възможно [19], [20], [38], [118].

Обектите за декомпозиция могат да се представят чрез графови модели, Тези модели се изграждат на базата на теорията на графите и тяхното приложение. Така задачата за декомпозиция на графовия модел се привежда към задача за рационалното му разделяне на подграфи, докато е необходимо или възможно. Това е оптимизационна задача, чието решение позволява отделянето на функционално обособени модули [19], [20], [21], [30].

Граф е всяка наредена двойка $G(V,E)$, в която $V=\{v_1,v_2,\ldots,v_n\}$ е множество, чиито елементи се наричат върхове, а $E=\{e_1,e_2,\ldots,e_m\}$ – множество, чиито елементи се наричат ребра.

Представянето на графовите модели е разгледано подробно в [102] и [108]. В настоящия дисертационен труд са използвани матрици и списъци на съседство.

Най-често използваните критерии за декомпозиция на графовите модели са:

- Минимална свързаност на подграфите;
- Минимален брой външни върхове;
- Определен брой върхове в подграфите и др.

Математическото формулиране на задачата за декомпозиция на графа G(V,E) е следната:

Даден е графът $G(V,E)$. Да се раздели на $N+1$ подграфи $G_1, G_2, G_3,\ldots,G_N, G_c$, където $G_p=(V_p,E_p)$, $p=1,2,3,\ldots,N$ са отделените подграфи, а $G_C=(V_C,E_C)$ е множеството на външните връзки между отделените подграфи. Търси се такова разделяне на графа G на подграфи, при което се удовлетворяват следните условия:

- Броят външни връзки да е минимален:

$$V_C = min(V_C),\ E_C = min(E_C);$$

- Всеки подграф да е различен от нулевия: $\forall G_p \neq 0,\ p=1,2,\ldots N,\ G_C \neq 0;$
- Обединението на всички подграфи да съвпада с изходния граф:

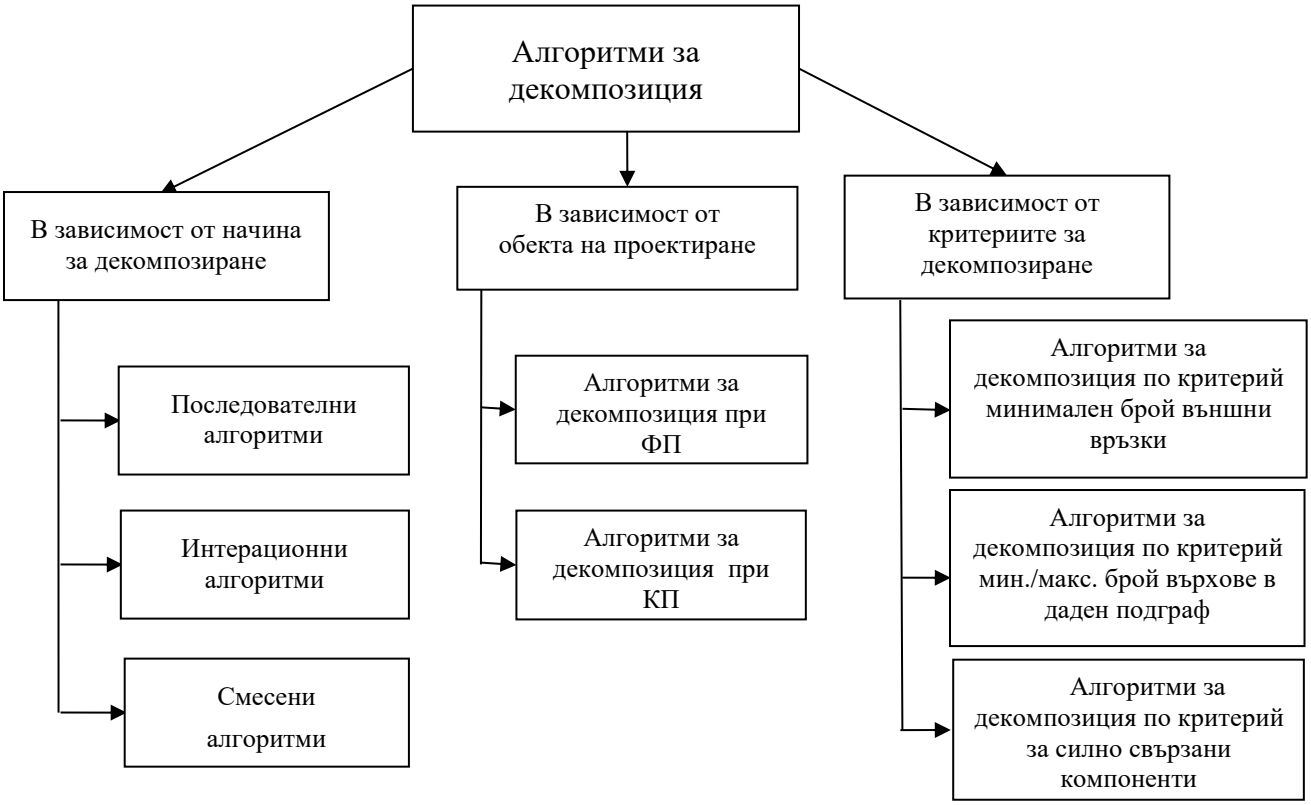
$$(\cup\ G_p) \cup G_C = G,\ p=1,2,\ldots,N;$$

Като външни връзки се разглеждат връзките (ребра или върхове) между отделните подграфи, а като вътрешни – връзките в отделните подграфи. Реброто E_i е вътрешно, ако $E_i \in E_p$, в противен случай се счита за външно. Същото важи и за върховете. Върхът V_i е вътрешен, ако $V_i \in V_p$, в противен случай е външен.

Обект на изследване на настоящата разработка са следните последователни алгоритми за декомпозиция на графови модели [19], [20], [21], [30]:

- алгоритъм с добавяне (**Д**);
- алгоритъм с отделяне (**О**);
- алгоритъм с припокриване (**П**);
- алгоритъм с неприпокриване (**Н**);

Алгоритмите за декомпозиция могат да се разделят на групи според определени зависимости: начин на разделяне; определящ критерий; етапи на проектиране и др. На *фиг.1.6* е представена класификация на тези алгоритми.



Фиг.1.6. Класификация на алгоритмите за декомпозиция

Изводи към Глава 1

- В проучените източници, цитирани в настоящата разработка, са описани различните алгоритми за декомпозиция, но липсва класификация за тях;
- Съществува методика за определяне на сложност на алгоритъм, но липсва такава за алгоритмите за декомпозиция;
- Повечето алгоритми за декомпозиция се разглеждат върху графови модели, които невинаги представят реални обекти;
- Липсват правила за избор на даден алгоритъм при решаване на конкретни задачи;
- Не е направено пълно изследване на алгоритмите за декомпозиция.

Глава 2. СЛОЖНОСТ НА АЛГОРИТМИТЕ ЗА ДЕКОМПОЗИЦИЯ

В тази глава са разгледани видовете сложност, както и различните форми на асимптотичните нотации. Определени са правила за изчисляване на сложност. Те са приложени върху процедурните стъпки на алгоритмите за декомпозиция. Чрез математически преобразования са изведени аналитични изрази и е определена оценката за сложност по време за всеки от изследваните алгоритми. Изчислени са максималния брой операции за декомпозиционните алгоритми.

2.1. Сложност на алгоритъм.

Най-често сложността се разделя на сложност по време и сложност по памет. Сложността по време се измерва в бързодействието на изследвания алгоритъм. То се изчислява като процесорно време чрез *формула (2.1)*:

$$T = \frac{N}{P} \qquad (2.1)$$

в която:

N – брой операции, необходими за изпълнението на алгоритъма;

P – брой операции, които процесора изпълнява за 1 сек.

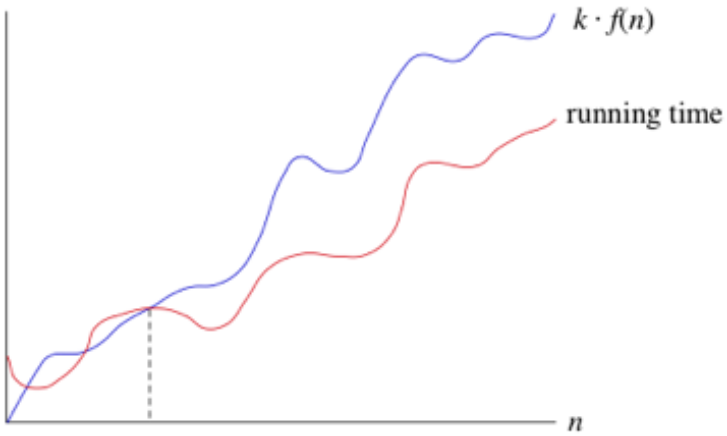
За всеки процесор параметърът P е известен. За да се оцени сложността на един алгоритъм, то трябва да се определи параметърът N. Това може да се направи по някои от следните случаи:

- ***Най-лош случай (worst-case)*** – този случай дава информация за максималния брой операции, необходими за изпълнението на алгоритъма. Той е най-често използваният;
- ***Среден случай (average-case)*** – този случай дава информация за средния брой на операциите;
- ***Най-добър случай (best-case)*** – този случай дава информация за минималния брой операции, необходими за изпълнението на алгоритъма.

Сложността на алгоритмите може да се определи чрез асимптотична нотация [39], [53], [63], [64], [66]. Тя представлява изменение на поведението на даден алгоритъм при различни входни данни. Времето за работа на един алгоритъм се разглежда като функция, зависима от размера на входа. Асимптотичната нотация фокусира вниманието си върху това как се изменя тази функция с увеличаване на размера на входните данни.

Съществуват няколко форми на асимптотични нотации: *Big-θ нотация, Big-O нотация, Big-Ω нотация.* Те ограничават времето на функцията

отгоре или отдолу, като посочват максималното и минималното време. В настоящата разработка е взет под внимание най-лошият случай. Той ни дава информация за максималното време, което може да се получи при работа на един алгоритъм. Това съответства на ограничение отгоре на функцията, отразяваща времето за изпълнение. *Big-O нотацията* прави точно това ограничение. На *фиг.2.3*. е показана нейната диаграма.



Фиг.2.3. Диаграма за Big-O нотация

Сложността по време бива следните видове: константна *O(1)*, логаритмична *O(log n)*, линейна *O(n)*, квадратична $O(n^2)$, кубична $O(n^3)$, експоненциална $O(2^n)$ и др. [27], [28], [36], [37], [39], [53], [58], [64], [91]. Кубичната и експоненциалната сложност са едни от най-бавните.

Под изчисляване на сложност се разбира броят на необходимите операции за изпълнение на алгоритъма. При изчисляването ѝ за една операция на алгоритъма се приемат:

- Всяко дефиниране и присвояване на стойност на променлива;
- Всеки вид сравняване;
- Всяка аритметична операция;
- Всяко добавяне или вземане на елемент от масив (множество, граф и т.н.);
- Всяко извикване на метод или функция.

2.2. Оценки по сложност на алгоритмите за декомпозиция

В този раздел е направена аналитична оценка на разгледаните алгоритми за декомпозиция. Използвани математически похвати върху процедурните им стъпки.

2.2.1.Определяне на сложност на алгоритъма с отделяне

За алгоритъма с отделяне най-лош случай има тогава, когато графът се раздели на максимален брой подграфи. Сложността на съответната стъпка е отбелязвана с буквата **O** и долен индекс, започващ с „о“ и приключващ с номера на съответната стъпка. За сравнение е направена и асимптотична нотация на алгоритъма.

Първоначално се получават оценките за всяка една стъпка на алгоритъма. Това става, като се направят анализи върху тях и се приложат горните правила. Оценките за всяка една стъпка са отразени в дисертационния труд с *формули* от *(2.2)* до *(2.9)*. Чрез математически преобразувания към тях и тяхното сумиране се получава *формула (2.10)*, с която може да се изчисли максималният брой операции за този алгоритъм.

$$O_{\text{о}} = \frac{7}{6}n^3 + 6n^2 - \frac{25}{6}n - 2 \qquad (2.10)$$

Като се приложи асимптотичната нотация върху *формула (2.10)* се вижда, че сложността на алгоритъма е кубична $O(n^3)$.

2.2.2.Определяне на сложност на алгоритъма с добавяне

За най-лош случай при алгоритъма с добавяне се разглеждат следните два сценария:

- I-ви сценарий – когато броят на върховете в отделните подграфи е 1, т.е. m=1.
- II-ри сценарий – когато броят на върховете в отделните подграфи е 2, т.е. m=2.

Сложността на съответната стъпка е отбелязвана с буквата O и долен индекс, започващ с „д“, придружен с номер за сценарий. За I-ви сценарий максималният брой операции на алгоритъма може да се изчисли чрез *формула(2.11)*.

$$O_{\text{д}I} = \frac{n^3}{3} + n^2 + \frac{20}{3}n - 7 \qquad (2.11)$$

За II-ри сценарий оценката на алгоритъма зависи от това дали *n* приема четна или нечетна стойност. Максималният брой операции при този сценарий се изчислява чрез *формули (2.25)* и *(2.26)*.

$$O_{\text{д}II\text{ч}} = \frac{1}{6}n^3 + \frac{5}{2}n^2 + \frac{16}{3}n - 21 \qquad (2.25)$$

$$O_{\text{д}II\text{н}} = \frac{1}{6}n^3 + \frac{5}{2}n^2 + \frac{16}{3}n - 11 \qquad (2.26)$$

След прилагането на асимптотичната нотация върху *формули (2.11), (2.25)* и *(2.26)* се вижда, че сложността на алгоритъма е кубична $O(n^3)$. При заместване на *n* с различни стойности се получава факта, че сценарий I-ви е най-лошият случай за този алгоритъм.

2.2.3.Определяне на сложност за алгоритъма с припокриване

Сложността на съответната стъпка се отбелязва с буквата О и долен индекс, започващ с „п“. Оценките за всяка една стъпка са отразени в дисертационния труд с *формули* от *(2.27)* до *(2.35)*. Чрез математически преобразувания към тях и тяхното сумиране се получава *формула (2.36)*, с която може да се изчисли максималният брой операции за този алгоритъм,

$$O_{\text{п}} = \frac{2}{3}n^3 + \frac{3}{2}n^2 + \frac{47+2z}{6}n + 1 \qquad (2.36)$$

в който z е броят на общите върхове.

След прилагането на асимптотичната нотация върху *формула (2.36)* се вижда, че сложността на алгоритъма е кубична $O(n^3)$.

2.2.4.Определяне на сложност за алгоритъма с неприпокриване

Сложността на съответната стъпка е отбелязвана с буквата О и долен индекс, започващ с *„н“*. Оценките за всяка една стъпка са отразени в дисертационния труд. Максималният брой операции на алгоритъма може да се изчисли чрез *формула (2.37),*

$$O_{\text{н}} = \frac{2(k+1)}{6}n^3 + \frac{3(k+2)}{2}n^2 + \frac{13}{2}n + \frac{k+6z+7}{6} \qquad (2.37)$$

в която:

k – е броят на подграфите;

z – е броят на външните върхове

След прилагането на асимптотичната нотация върху *формула (2.37)* се вижда, че сложността на алгоритъма е кубична $O(n^3)$.

2.3. Анализ на получените резултати:

С математически преобразувания са определени оценките за сложност на представените в първа глава алгоритми за декомпозиция. Те са изследвани с различни графови модели с брой върхове от 14 до 600. Чрез *формули (2.10), (2.11), (2.36)* и *(2.37)* е определен максималният брой операции за всеки от тях. Получените резултати са представени в *табл. 2.1* и *2.2*.

Табл.2.1. *Оценка на алгоритмите с добавяне и отделяне*

N	**$O_О$**	**$O_Д$**
14	4 317	1 197
22	15 233	4 173
63	315 271	87 731
100	1 226 248	343 993
600	254 157 498	72 363 993

От *табл. 2.1* се забелязва, че за алгоритъма с добавяне максималният брой операции е по-малък от алгоритъма с отделяне.

Табл.2.2. *Оценка на алгоритмите с припокриване и непripокриване*

N	**$O_П$**	**$O_Н$**
14	2 334	1 075
22	7 998	3 887
63	173 146	85 600
100	682 451	338 754
600	144 544 701	72 182 504

От *табл. 2.2* се забелязва, че за алгоритъма с непripокриване максималният брой операции е по-малък от алгоритъма с припокриване.

Изводи към Глава 2

- За графови модели, които се разделят по най-лошия сценарий, алгоритъмът с добавяне е по-добър от алгоритъма с отделяне.
- За графови модели, които се разделят по най-лошия сценарий, алгоритъмът с непripокриване е по-добър от алгоритъма с припокриване.
- С нарастване броят на върховете многократно нараства и броят на използваните операции

Глава 3. ИЗСЛЕДВАНЕ НА АЛГОРИТМИ ЗА ДЕКОМПОЗИЦИЯ.

В тази глава е направено аналитично изследване на алгоритмите за декомпозиция. Те са приложени за декомпозиция на различни по размер

графови модели. На базата на получените резултати са обобщени изводи за приложимостта на изследваните алгоритми.

Цел и задачи на изследването

Целта на изследването е да се направи съпоставка на алгоритмите за декомпозиция. Като обект на декомпозиция се разглеждат графови модели с различни размери (различен брой елементи/върхове).

За целите на изследването са формирани следните задачи:

ЗАДАЧА 1:

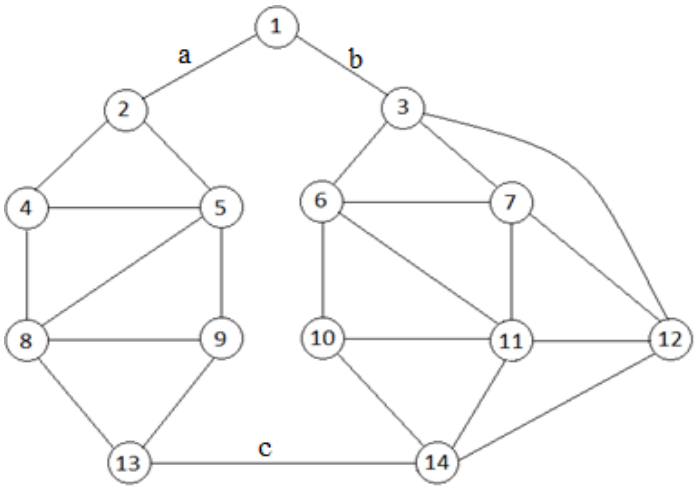
Нека е даден граф *G(V,E)* с определен брой върхове и ребра между тях. Да се раздели графа на подграфи при определящ критерий минимален брой външни ребра между подграфите. Допълнително условие е броят на върховете във всеки подграф да бъде по-малък или равен на m_{max}.

ЗАДАЧА 2:

Нека е даден граф *G(V,E)* с определен брой върхове и ребра между тях. Да се раздели графът на части при определящ критерий минимален брой външни върхове между подграфите. Допълнително условие е броят на върховете във всеки подграф да бъде по-малък или равен на m_{max}.

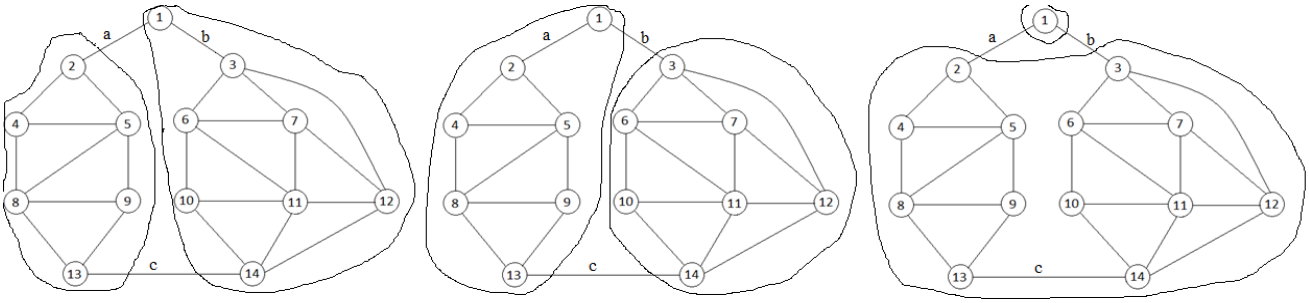
За решаването на задача 1 са използвани алгоритмите с добавяне и отделяне, а за задача 2 – алгоритмите с припокриване и неприпокриване. В тази глава е разгледан подробно процесът на декомпозиция на графови модели при използване на съответните алгоритми.

На *фиг.3.1* е даден графов модел с 14 върха за разделяне.



Фиг.3.1. Граф за декомпозиране с 14 върха

Без допълнителното условие за m_{max} на задача 1, този граф се разделя на два подграфа по три възможни начина. Те са показани на *фиг.3.2*.

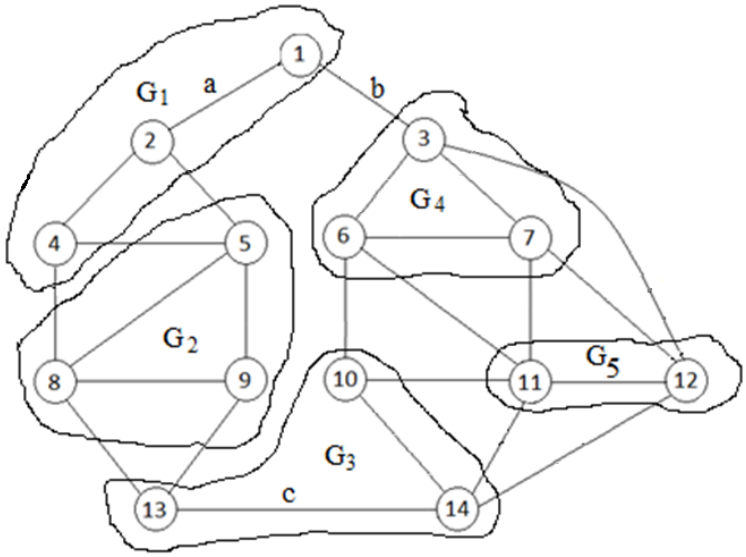


а) *б)* *в)*

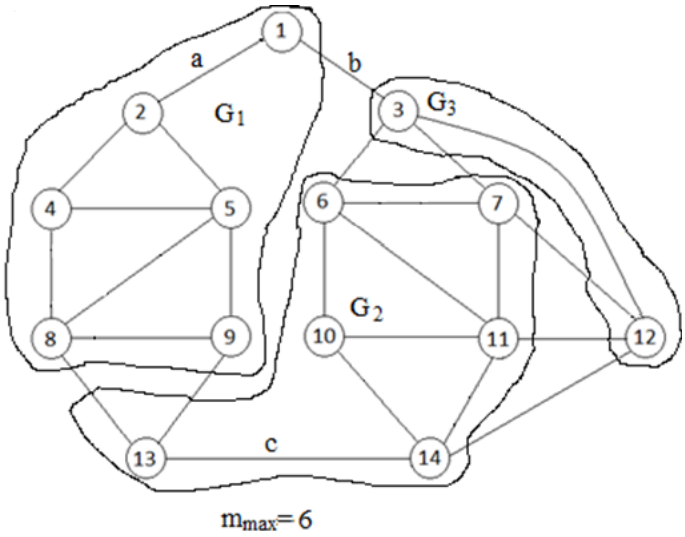
Фиг.3.2. Възможни варианти за разделяне на графа

Ако е необходимо отделните подграфи да имат приблизително еднакъв брой върхове, то се поставя допълнително условие за m_{max}. В този случай най-добрият вариант на разделяне е показан на *фиг.3.2 б)*. В зависимост от стойността за m_{max} графът може да се раздели и на повече подграфи, но това ще увеличи броя на външните връзки, което противоречи на зададения критерий.

За разделянето на графа от *фиг.3.1* за задача 1 се използват алгоритмите с отделяне и добавяне. В зависимост от стойността на m_{max}, алгоритъмът с добавяне разделя графовия модел по различен начин. На *фиг.3.3* е показано разделянето за $m_{max}=3$, а на *фиг.3.6* за $m_{max}=6$.

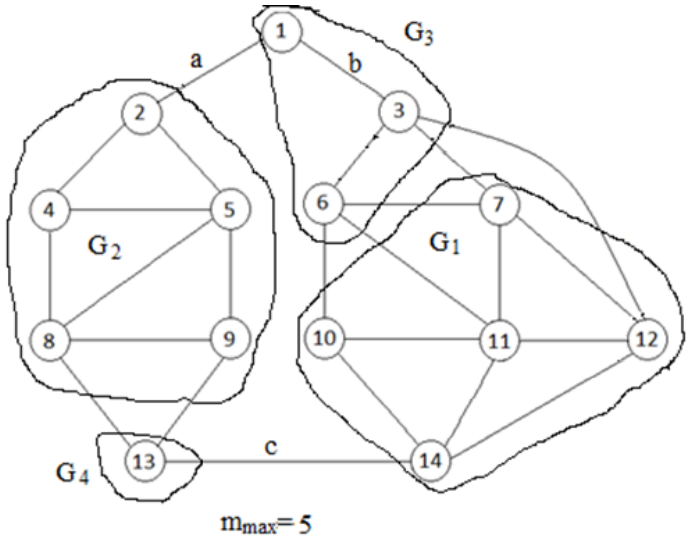


Фиг.3.3. Разделяне на графа при $m_{max}=3$ по алгоритъма с добавяне.

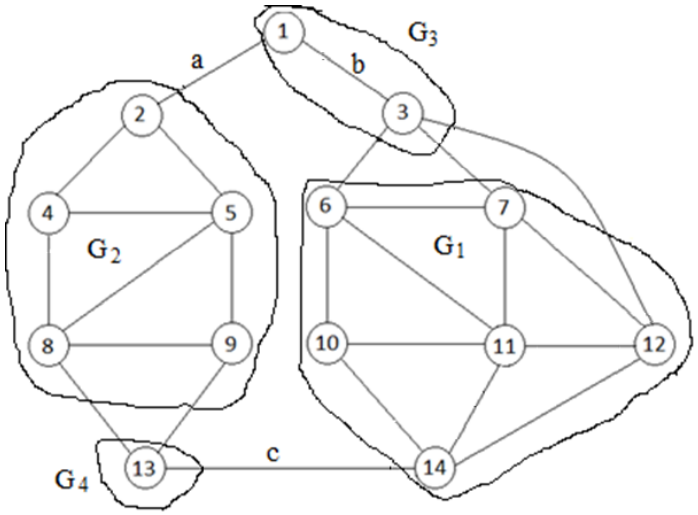


Фиг.3.6. Разделяне на графа при m_{max}=6 по алгоритъма с добавяне.

При алгоритъма с отделяне графовият модел се разделя на различни подграфи в зависимост от стойността на m_{max}, но при определени стойности някои от подграфите са еднакви. На *фиг.3.9* и *фиг.3.10* е показано разделяне спрямо алгоритъма с отделяне при m_{max}=5 и m_{max}=6. От тях се забелязва, че подграфът G_2 е един и същи за двете декомпозиции.



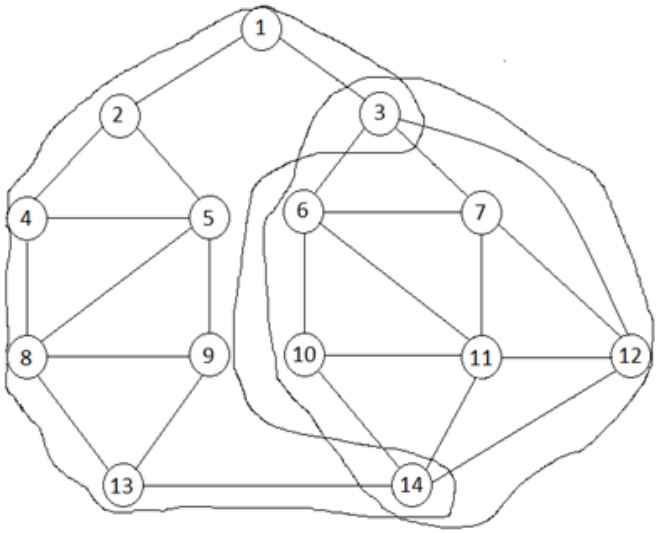
Фиг.3.9. Разделяне на графа при m_{max}=5 по алгоритъма с отделяне.



Фиг.3.10. Разделяне на графа при m_{max}=6 по алгоритъма с отделяне.

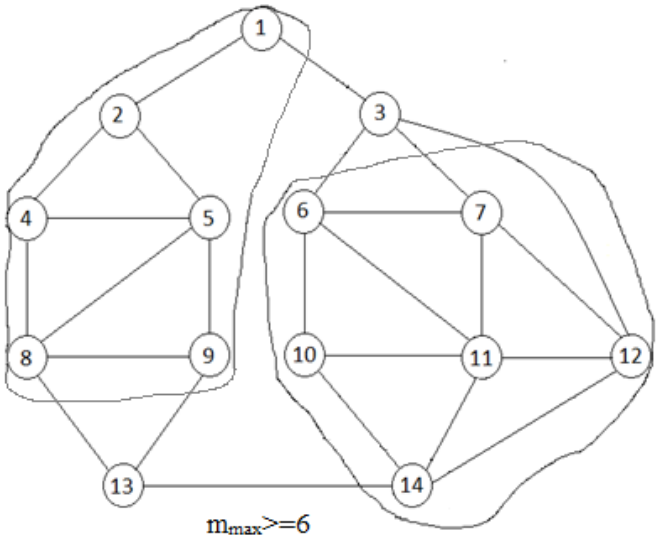
За решаването на задача 2 от трета глава се използват алгоритмите с припокриване и неприпокриване. Алгоритъмът с припокриване отделя общи върхове, а алгоритъмът с неприпокриване – външни.

На *фиг.3.11* е показано графичното разделяне на графа по алгоритъма с припокриване. Този алгоритъм не зависи от стойността на m_{max}, за разлика от алгоритъма с неприпокриване.

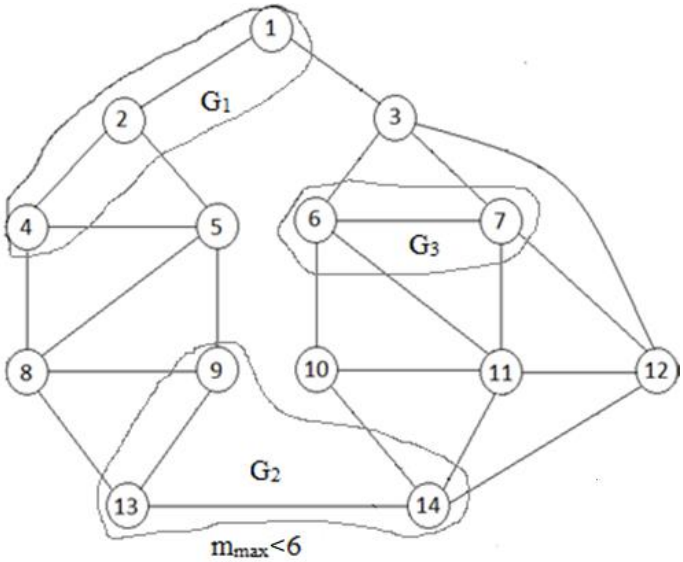


Фиг.3.11. Разделяне на графа по алгоритъма с припокриване.

При разделянето на графовия модел чрез алгоритъма с неприпокриване се отделят външни върхове. Техният брой, както и броят на отделните подграфи, зависи от стойността на m_{max}. За графа от *фиг.3.1* съществено влияние оказва m_{max}, когато е по-голямо или по-малко от 6. Това се вижда на *фиг.3.12* и *фиг.3.13*.



Фиг.3.12. Разделяне на графа по алгоритъма с непripокриване при $m_{max} \geq 6$.



Фиг.3.13. Разделяне на графа по алгоритъма с непripокриване при $m_{max} < 6$.

В глава 3 на дисертационния труд задачите са решени по изследваните алгоритми и за графов модел с 22 върха.

Анализ на получените резултати.

При сравняване на резултатите от разделянето по алгоритъма с добавяне се установява, че най-доброто разделяне невинаги е при реципрочни стойности на m_{max}, спрямо n (броят на върховете на графовия модел). Например, за разделянето на графовия модел на два подграфа е необходимо да се подбере m_{max}, да бъде приблизително равно на $n/2$.

Декомпозирането на графовия модел от *фиг.3.1* при $m_{max}=7$ е най-доброто по зададения критерий.

Алгоритъмът с отделяне разделя графовия модел на повече от два подграфа, при $m_{max}<\frac{2n}{3}$. При алгоритъма с добавяне такова решение се получава при $m_{max}<\frac{n}{2}$.

Алгоритъмът с отделяне разделя графовия модел на приблизително еднакви по големина подграфи, с малки изключения. Възможно е да има подграфи с по един връх. В зависимост от обекта, който е представен чрез графовия модел, това разделяне може да няма логически смисъл.

Алгоритъмът с припокриване винаги разделя графовия модел на толкова подграфи, колкото е необходимо. Не зависи от фактора m_{max}.

Намаляването на m_{max} води до увеличаване на броя на външните върхове.

Изводи към Глава 3

- При алгоритъма с отделяне m_{max} не указва влияние в разделянето на графовия модел за стойности в интервала $(\alpha_{max}+1, n-n_{G1})$.
- Алгоритъмът с отделяне е по-подходящ при декомпозиране на графовия модел на повече части.
- Алгоритъмът с неприпокриване е по-подходящ при контролирано разделяне на графовите модели.

Глава 4: ПРОГРАМНИ ПРИЛОЖЕНИЯ ЗА ДЕКОМПОЗИЦИЯ НА ГРАФОВИ МОДЕЛИ

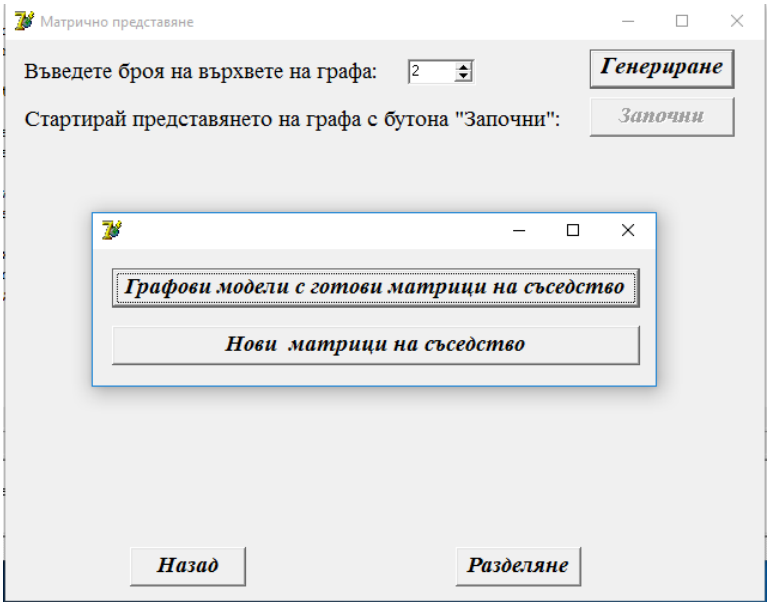
В тази глава са представени разработените програмни приложения за целите на проведеното изследване. Приложенията са реализирани чрез програмните среди Borland Delphi 7 и MATLAB. С тях са изследвани представените в глава първа алгоритми. Получените резултати са представени и използвани за сравнение на изследваните алгоритми.

4.1. Програмни приложения.

Алгоритмите с добавяне и отделяне са реализирани като програмно приложение, което включва следните основни функции:

- Представяне на графовия модел;
- Избор на алгоритъм за декомпозиция;
- Избор на максимален брой върхове в един подграф;
- Разделяне на графовия модел.

Като входни данни за приложението се използват матрично и списъчно представяне на графовия модел. За улеснение е създадена форма (*фиг.4.2*), която позволява въвеждането на графовия модел от файл.



Фиг.4.2. Зареждане на графовия модел от файл

Като изходни данни за приложението се извеждат:

- Списъкът с върховете в отделните подграфи;
- Времето за изпълнение на съответния алгоритъм;
- Броят на използваните операции от алгоритъма.

Програмните приложения за алгоритмите с припокриване и неприпокриване са реализирани като скриптови файлове. Работата с тях е показана от *фиг.4.19* до *фиг.4.23*.

За входни данни се използва графов, модел представен чрез:

- Вектор с върховете;
- Матрицата на съседство;
- Максималният брой върхове в един подграф.

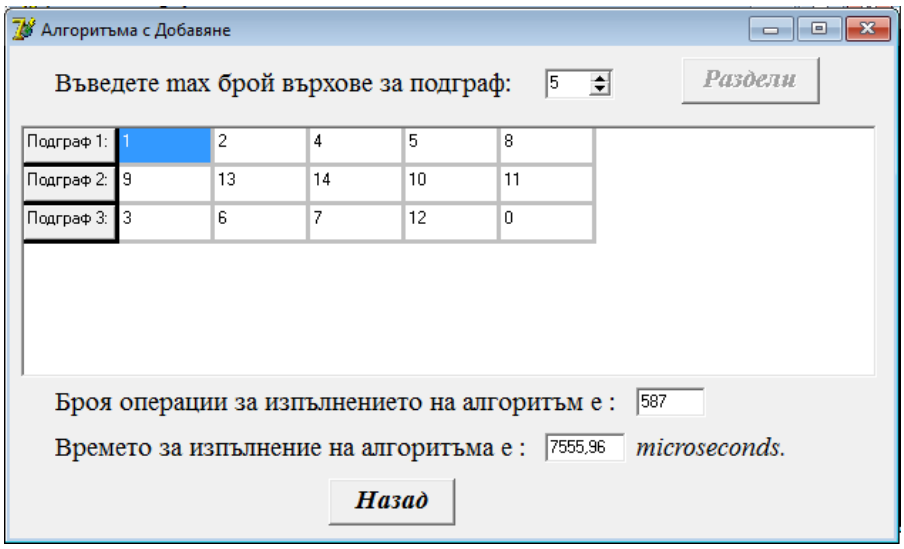
За изходни данни се извеждат:

- Списък с върховете в отделните подграфи;
- Списък с общите или външните върхове;
- Броят на използваните операции;
- Времето за изпълнение на съответния алгоритъм за разделяне.

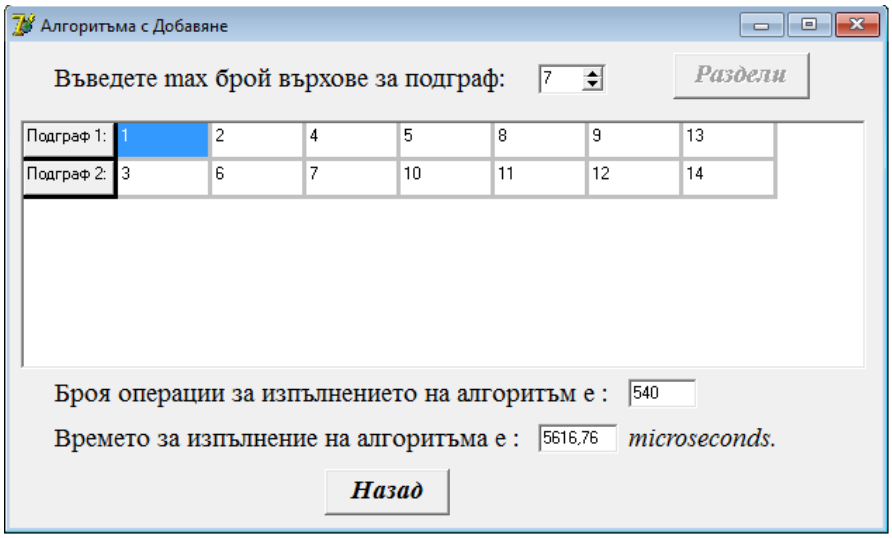
4.2. Изследване на алгоритмите чрез програмните приложения

Изследването на алгоритмите с това приложение е направено върху различни графови модели с различен брой върхове. В тази точка са показани резултати от разделянето на графовите модели, представени в глава 3.

На *фиг.4.10* и *фиг.4.11* е визуализиран резултат от разделянето на графовия модел с 14 върха по алгоритъма с добавяне чрез програмното приложение. На *фиг.4.10* е показано разделянето при $m_{max}=5$, а на *фиг. 4.11* – при $m_{max}=7$.



Фиг. 4.10. Разделяне чрез алгоритъма с добавяне за графа с 14 върха при $m_{max}=5$

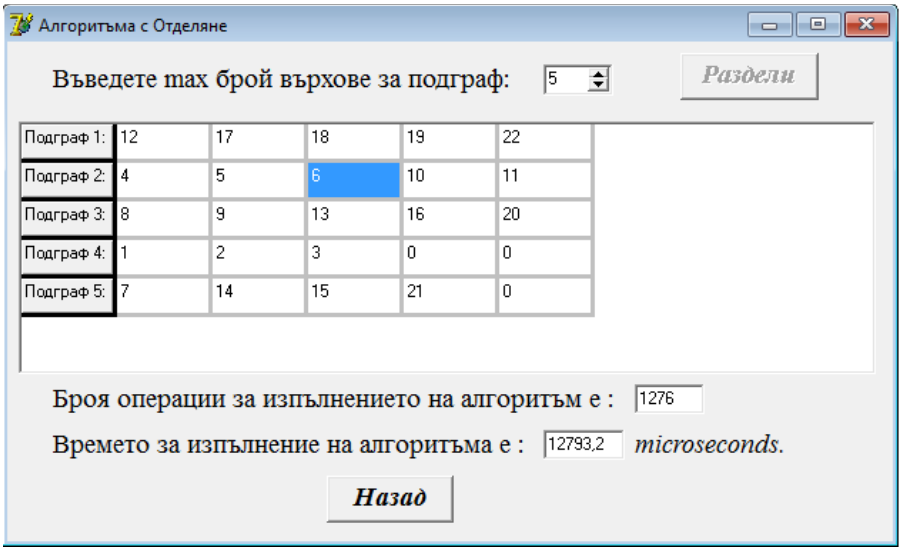


Фиг. 4.11. Разделяне чрез алгоритъма с добавяне за графа с 14 върха при $m_{max}=7$

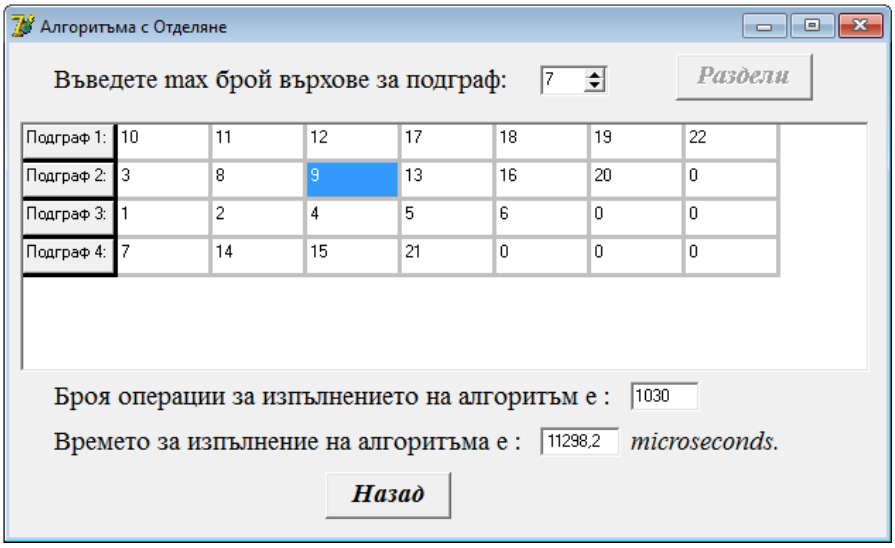
Забелязва се, че при повишаване на m_{max}, броят на отделните подграфи намалява, което води до намаляване на броя на използваните операции и времето за изпълнение.

От *фиг.4.12* до *фиг.4.18* от дисертационния труд са представени част от резултатите при разделяне на графовия модел с 22 върха по алгоритъма с отделяне. Декомпозирането е извършено при различни стойности на m_{max}.

На *фиг.4.14* е визуализиран резултатът от разделянето на графовия модел при m_{max}=5, а на *фиг.4.15* при m_{max}=7.



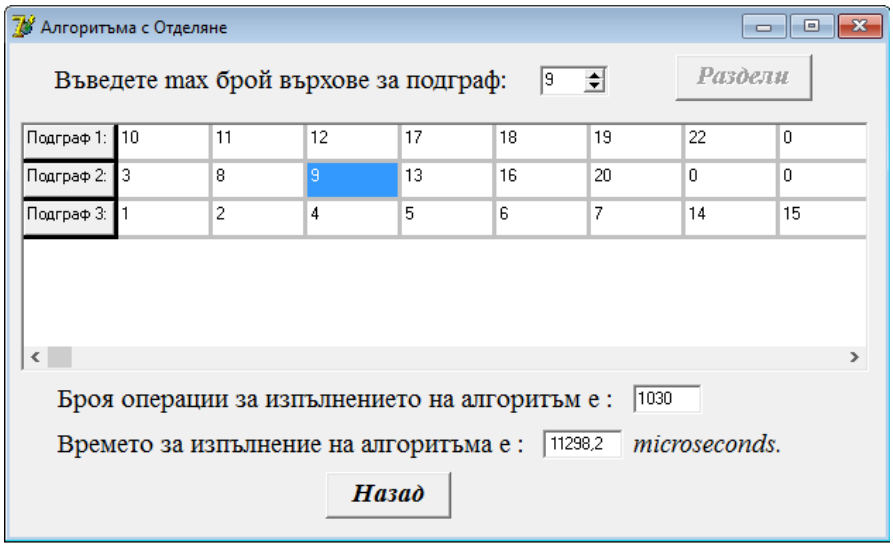
Фиг. 4.14. Разделяне чрез алгоритъма с отделяне за графа с 22 върха при m_{max}=5



Фиг. 4.15. Разделяне чрез алгоритъма с отделяне за графа с 22 върха при m_{max}=7

При m_{max}=5, графовият модел се разделя на 5 подграфа, а при m_{max}=7 – на 4 подграфа. Забелязва се, че последният подграф и в двата случая (*фиг.4.14* и *фиг.4.15*) съдържа едни и същи върхове (7, 14, 15 и 21).

На *фиг.4.16* е визуализиран резултатът от разделянето на графовия модел при m_{max}=9.



Фиг. 4.16. Разделяне чрез алгоритъма с отделяне за графа с 22 върха при $m_{max}=9$

За алгоритъма с отделяне се забелязва факта, че при $m_{max}=7$ и $m_{max}=9$, разгледаният графов модел с 22 върха се разделя по един и същ начин, поради което броят на използваните операции и времето за изпълнение са еднакви.

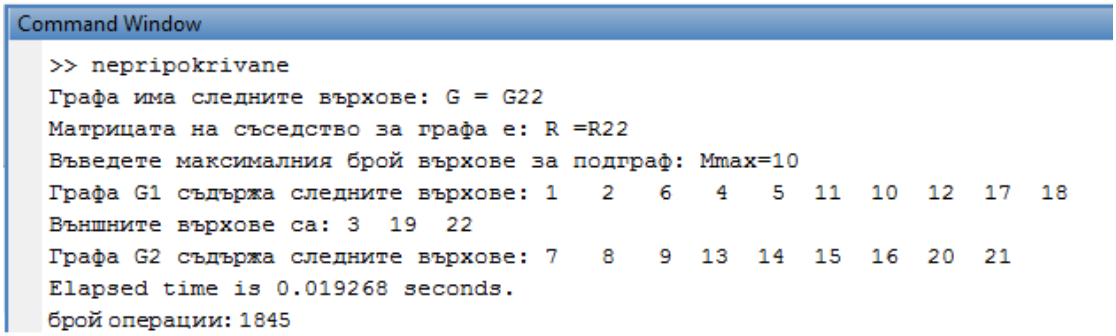
На *фиг.4.19* са представени резултатите при разделяне на графовия модел с 14 върха по алгоритъма с припокриване, получени от програмното приложение.

```
Command Window
>> pripokrivane
Графа има следните върхове: G = G14
Матрицата на съседство за графа е: R =R14
Общите върхове между подграфи G1 и G2 са: 3  14
Графа G1 съдържа следните върхове: 1   2   4   5   8   9  13   3  14
Графа G2 съдържа следните върхове: 3   7  12   6  10  11  14
Elapsed time is 0.017616 seconds.
брой операции: 950
```

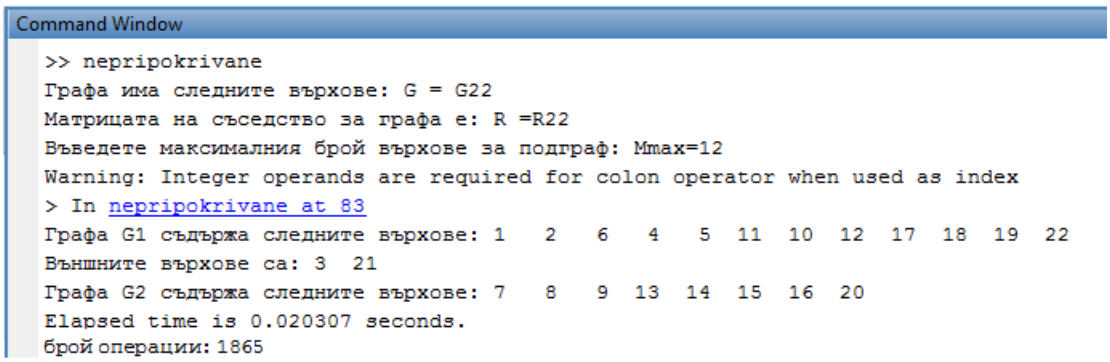
Фиг. 4.19. Разделяне чрез алгоритъма с припокриване за графа с 14 върха

От *фиг.4.20* до *фиг.4.23* са представени част от резултатите при разделяне на графовия модел с 22 върха по алгоритъма с неприпокриване. Декомпозирането е извършено при различни стойности на m_{max}.

На *фиг.4.22* и *фиг.4.23* е визуализиран резултатът от разделянето на графовия модел при $m_{max}=10$ и $m_{max}=12$. И при двата варианта графовият модел се разделя на два подграфа, но при $m_{max}=10$ броят на външните върхове е 3, а при $m_{max}=12$ – е само 2 върха.



Фиг. 4.22. Разделяне чрез алгоритъма с неприпокриване за графа с 22 върха при m_{max}*=10*



Фиг. 4.23. Разделяне чрез алгоритъма с неприпокриване за графа с 22 върха при m_{max}*=12*

4.3. Анализ на получените резултати

Разработените приложения са използвани за решаване на поставените задачи в Глава 3. Извършени са изследвания върху графови модели с различни размери, с брой на върховете от 14 до 600 и при различни стойности на m_{max}. Получените резултати при решаване на първата задача по алгоритмите с добавяне и отделяне са представени от *табл.4.1* до *табл.4.5*.

В първата колона на всяка от таблиците са посочени различни стойности за m_{max} за съответния граф. Означението на буквите в тях е следното:

- N – брой на използваните операции за съответния алгоритъм спрямо приложението;
- T – време за изпълнение на съответния алгоритъм в милисекунди *ms*;
- n – брой на отделните подграфи след прилагането на алгоритъма;
- K – брой на външните връзки;

***Табл.4.1**. Резултати за алгоритмите с добавяне и отделяне за графов модел с 14 върха*

m_{max}	С добавяне				С отделяне			
	N	**T (*ms*)**	**n**	**K**	**N**	**T (*ms*)**	**n**	**K**
3	642	7,90	5	14	632	7,45	5	13
4	624	7,64	4	11	514	6,70	4	11
5	587	7,55	3	9	412	5,61	4	9
6	750	7,61	3	8	389	6,01	4	7
7	540	5,61	2	2	389	6,02	4	7
8	657	5,82	2	4	273	3,24	2	4
9	819	6,01	2	5	273	3,24	2	4

***Табл.4.2.** Резултати за алгоритмите с добавяне и отделяне за графов модел с 22 върха*

m_{max}	С добавяне				С отделяне			
	N	**T (*ms*)**	**n**	**K**	**N**	**T (*ms*)**	**n**	**K**
4	1686	16,11	6	19	1507	12,37	8	21
5	1679	16,89	5	14	1276	12,79	6	17
6	1612	14,32	4	12	1090	12,11	6	16
7	1855	16,08	4	11	1030	11,29	5	13
8	1712	12,99	3	9	1030	11,29	5	13
9	1954	13,83	3	10	1030	11,29	5	13
10	2313	14,11	3	8	1030	11,29	5	13
11	1665	9,45	2	4	1030	11,29	5	13
12	1910	9,79	2	2	1030	11,29	5	13
15	2882	9,71	2	4	591	2,05	2	6

***Табл.4.3.** Резултати за алгоритмите с добавяне и отделяне за графов модел с 63 върха*

m_{max}	С добавяне				С отделяне			
	N	**T (*ms*)**	**n**	**K**	**N**	**T (*ms*)**	**n**	**K**
5	12345	87,90	13	31	11565	53,32	19	40
10	12356	87,72	7	21	10432	50,23	11	25
15	11865	86,34	5	9	8956	45,61	7	15
21	10986	70,63	3	3	8956	45,61	7	15
26	11457	76,16	3	7	8956	45,61	7	15
32	11195	73,82	2	4	8956	45,61	7	15
38	13457	74,01	2	5	8956	45,61	7	15
48	15325	75,35	2	9	2091	5,36	2	9

Табл.4.4. *Резултати за алгоритмите с добавяне и отделяне за графов модел със 100 върха*

m_{max}	С добавяне				С отделяне			
	N	**T (*ms*)**	**n**	**K**	**N**	**T (*ms*)**	**n**	**K**
10	60422	127,90	10	25	48132	95,45	14	53
30	58348	126,55	4	9	29812	76,61	11	21
40	57965	124,61	3	8	29812	76,61	11	21
53	52465	110,61	2	3	29812	76,61	11	21
64	61957	110,82	2	7	29812	76,61	11	21
83	69819	112,01	2	10	2765	4,23	2	12

Табл.4.5. *Резултати за алгоритмите с добавяне и отделяне за графов модел с 600 върха*

m_{max}	С добавяне				С отделяне			
	N	**T (*ms*)**	**n**	**K**	**N**	**T (*ms*)**	**N**	**K**
20	280654	857,90	30	84	95632	205,45	53	123
60	275123	827,64	10	32	79514	186,71	24	73
130	269865	819,55	5	16	79541	186,71	24	73
200	254563	795,61	3	10	79514	186,71	24	73
305	250540	775,61	2	6	79514	186,71	24	73
402	286657	781,82	2	11	79514	186,71	24	73
565	310819	786,01	2	23	4105	5,24	2	56

За решаването на втора задача от трета глава се използват алгоритмите с припокриване и неприпокриване. От *табл.4.6* до *табл.4.11* са представени резултатите на двата алгоритъма, при различните графови модели, отново с върхове от 14 до 600.

В първата колона на всяка от таблиците са посочени различни стойности на m_{max} за съответния граф. Означението на буквите в тях е следното:

- N – брой на използваните операции за съответния алгоритъм;
- T – време за изпълнение на съответния алгоритъм в милисекунди *ms*;
- n – брой на отделните подграфи след прилагането на алгоритъма;
- F – брой на граничните върхове (общи или външни);

Табл.4.6. Алгоритъмът с неприпокриване за графов модел с 14 върха

m_{max}	N	T (*ms*)	n	F
3	842	13,90	4	7
4	825	11,84	3	6
5	837	12,51	3	6
6	720	8,51	2	2
7	730	9,73	2	2
8	724	9,02	2	2

Табл.4.7. Алгоритъмът с неприпокриване за графов модел с 22 върха

m_{max}	N	T (*ms*)	n	F
4	2345	26,03	4	8
5	2452	28,06	4	8
6	2112	25,11	3	7
7	1885	20,28	3	6
8	1904	21,67	3	5
10	1845	19,26	2	3
11	1881	20,36	2	2

Табл.4.8. Алгоритъмът с неприпокриване за графов модел с 63 върха

m_{max}	N	T (*ms*)	n	F
10	15342	105,85	5	11
21	13624	94,64	3	7
32	12287	90,55	2	3
38	13550	91,79	2	4
48	13561	92,08	2	5

Табл.4.9. Алгоритъмът с неприпокриване за графов модел с 100 върха

m_{max}	N	T (*ms*)	n	F
10	80642	180,90	8	15
30	75624	175,64	3	11
40	72587	173,55	3	9
53	70750	170,61	2	2
64	72940	171,61	2	4
83	73257	172,82	2	7

Табл.4.10. Алгоритъмът с неприпокриване за графов модел с 600 върха

m_{max}	N	T (*ms*)	n	F
20	356242	950,90	28	41
60	330624	935,64	10	30
130	335987	909,55	5	35
200	310750	886,61	3	8
305	301540	865,61	2	4
402	305657	878,82	2	8
565	304819	870,01	2	6

Табл. 4.11. Резултати за алгоритъма с припокриване от 14 до 600 върха

Брой върхове	С припокриване			
	N	T (*ms*)	n	F
14	950	17,61	2	2
22	2705	38,64	2	2
63	12582	92,55	7	6
100	45250	150,61	10	9
600	90540	656,61	20	18

Получените резултати за времето на изпълнение и броят на използваните операции позволяват съпоставянето на двата алгоритъма (с припокриване и неприпокриване).

Изводи към Глава 4

От резултатите представени в т.4.3., могат да се направят следните изводи:

- При увеличаване стойността на m_{max}, времето за изпълнение на алгоритъма с добавяне намалява до достигане на минимално $\boldsymbol{K}$, след което се запазва приблизително еднакво.
- При декомпозиция на графови модели по алгоритъма с отделяне броят на използваните операции и времето за изпълнение намалява с увеличаване на m_{max}.
- По критерия минимален брой външни връзки алгоритъмът с добавяне е по-добър от алгоритъма с отделяне.
- Алгоритъмът с припокриване е по-бърз от алгоритъма с непripокриване.
- При декомпозиция на графови модели с големи размери е по-удачно да се използва алгоритъмът с припокриване, а при по-малко – алгоритъмът с непripокриване.

Глава 5: ПРИЛОЖЕНИЕ НА АЛГОРИТМИТЕ ЗА ДЕКОМПОЗИЦИЯ ПРИ ФУНКЦИОНАЛНО И КОНСТРУКТИВНО ПРОЕКТИРАНЕ

В тази глава са представени приложения на алгоритмите за декомпозиция при решаване на задачи от функционалното и конструктивното проектиране на схеми и системи. На базата на получените резултати са направени изводи за тяхната приложимост.

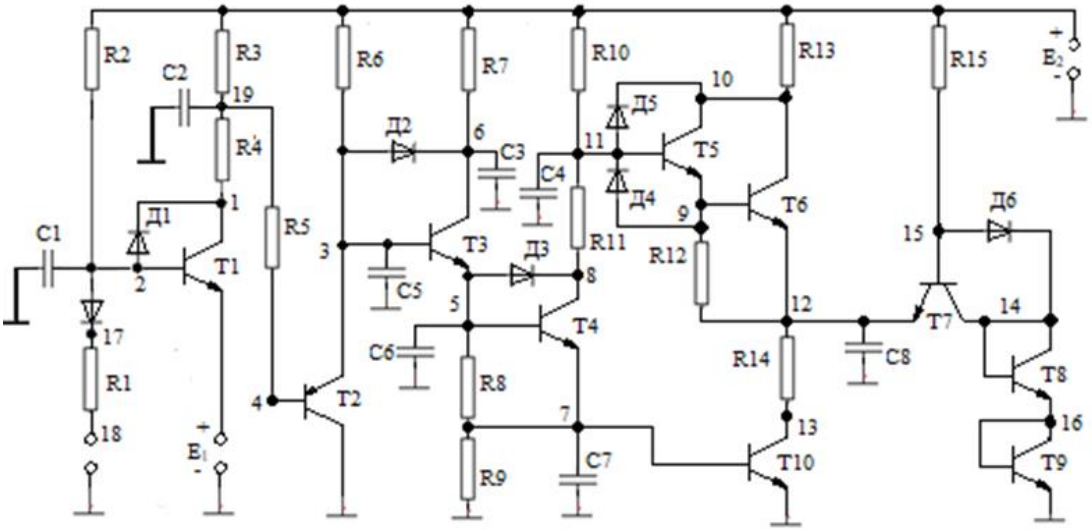
Приложение на алгоритмите за декомпозиция при функционално проектиране

Декомпозицията на една сложна система на отделни подсистеми, които са свързани с други функционални модули и изпълняват конкретни функции, се извършва по йерархичен принцип. Така например функционалните модули от най-ниско ниво (неделими функционални модули) са логическите елементи, отделните чипове памет, микропроцесорните схеми и др. Съвкупността от такива функционални модули образуват функционални модули от второ ниво (например микропроцесорен блок) и т.н.

Ако на множеството на възлите на системата (схема) се съпостави множеството на върховете V на графа $G(V,E)$, а на множеството на елементите на системата (схемата) - множеството на ребрата E, то задачата

за декомпозиране на системата (схемата) на части се привежда към задача за разделяне на графа G на слабосвързани помежду си подграфи G_1, G_2, G_3,....,G_N.

На *фиг.5.4* е представена принципна електронна схема на адресен формирователь. За по-лесното изследване на нейната проводимост е необходимо, тя да се раздели на обособени логически блокове, като се използват алгоритмите, представени в глава първа на дисертационния труд.

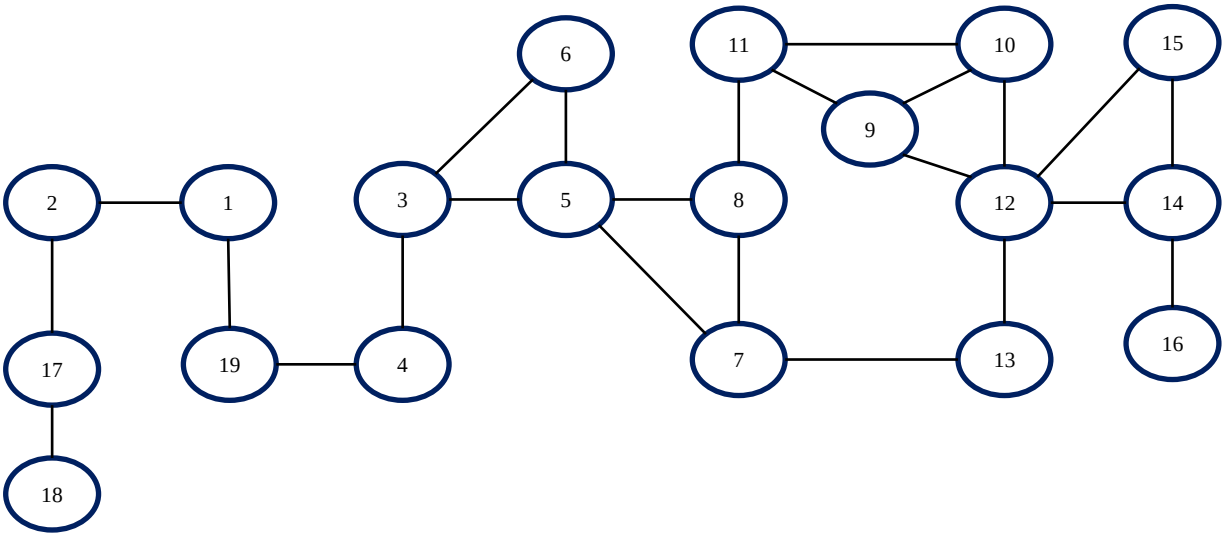


Фиг.5.4. Ел. схема на адресен формирователь.

За решаването на тази задача първоначално е построен графът *G(V,E)* за схемата на базата на следната аналогия: върховете *V* съответстват на потенциалите на схемата, а ребрата *E* – на отделните елементи между тях. След това от него е изключен базисният връх (върхът с най-голямо тегло).

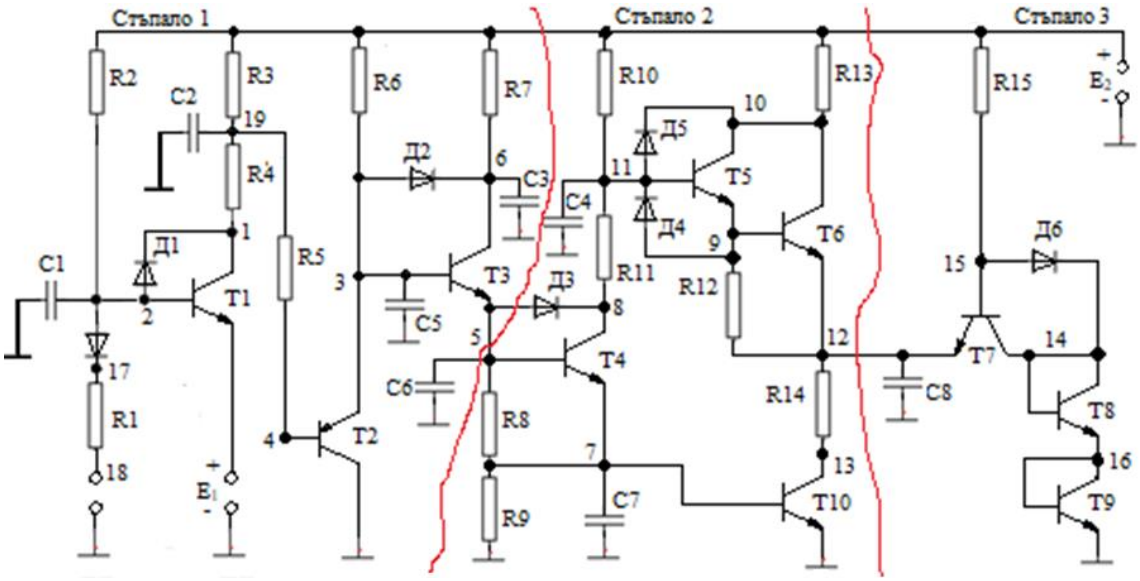
За разделянето на този графов модел трябва да се използват такива алгоритми, при които се получават външни или общи върхове. От разгледаните алгоритми такова разделяне извършват алгоритмите с припокриване и неприпокриване.

Броят на ребрата между отделните върхове не оказва влияние на работа на тези алгоритми. Поради тази причина графовият модел може да се опрости, като се премахнат паралелните ребра. Номерацията на върховете съответства на номерацията на потенциалите в схемата. Графовият модел на адресния формирователь от *фиг.5.4* е представен на *фиг.5.6.*



Фиг.5.6. Опростен вариант на графовия модел отразяващ структурата на схемата за адресен формирователь

Като се използва алгоритъма с припокриване за този граф, то той се разделя на три подграфа с общи върхове 5 и 12. След прилагането на декомпозицията върху схемата на адресния формирователь се получава следното разделяне, показано на *фиг.5.8*.

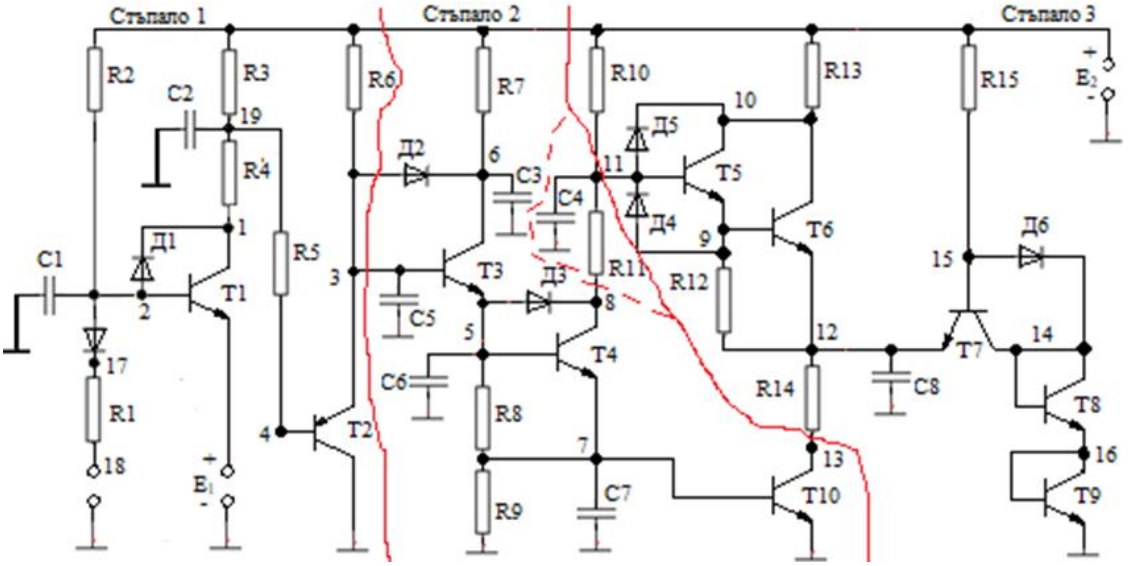


Фиг.5.8. Разделяне на схемата спрямо алгоритъма с припокриване

Схемата се разделя на три подсхеми, всяка от тях реализира отделни стъпала. Потенциал 5 се явява общ за стъпало 1 и 2. Той може да играе ролята на изход за стъпало 1 и вход за стъпало 2. По същата логика потенциал 12 се явява изходен за стъпало 2 и входен за стъпало 3.

Декомпозирането на графовия модел от *фиг.5.6* с използване на алгоритъма с неприпокриване е направено за различни стойности на m_{max} (максимален брой върхове в един подграф).

При $m_{max}=10$, графът се разделя на два подграфа, като *връх 5* е външен. Така схемата се разделя на две стъпала при потенциал *5*. При $m_{max}=6$ и $m_{max}=7$, графовият модел се разделя на три подграфа. Външните върхове и при двете разделяния са три, като два от тях са еднакви (3 и 13). При $m_{max}=6$, като трети външен се явява връх 11, а при $m_{max}=7$ – връх 8. На *фиг.5.13* е показано това разделяне на схемата, като разликата между отделните варианти е илюстрирана с пунктирана линия.



Фиг.5.13. Разделяне на схемата спрямо алгоритъма с неприпокриване

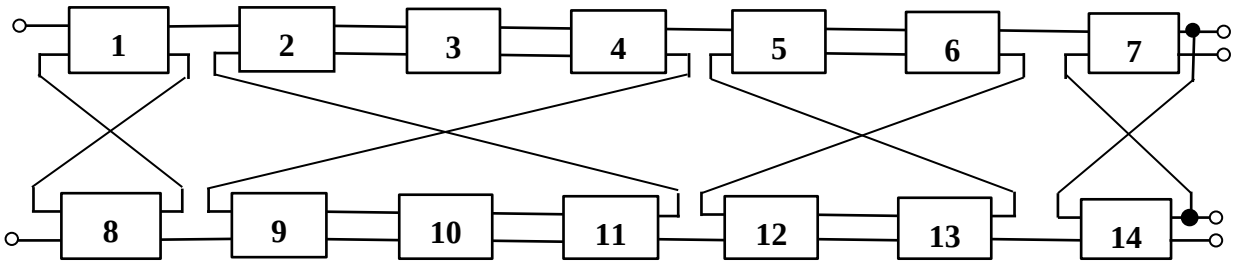
Приложение на алгоритмите за декомпозиране при конструктивно проектиране.

При конструктивно проектиране се решават задачите, свързани със синтез на конструкцията, верификация и документиране на конструктивния проект. Основна задача при синтез на конструкцията е задачата за компоновка на конструктивните модули по конструктивното поле. Тя включва решаването на три основни задачи: типизация, покритие и декомпозиция.

Задачата за компоновка се свежда до декомпозиране на системата на отделни функционални възли, които могат да се поместят в конструктивни модули от различни йерархични нива. Най-използваните критерии за решаване на тази задача са размерът на функционалния модул и броят на външните връзки между отделните функционални възли, разположени в различни конструктивни модули. Оптималното решение на задачата за декомпозиция е свързано с определянето на минимален брой модули.

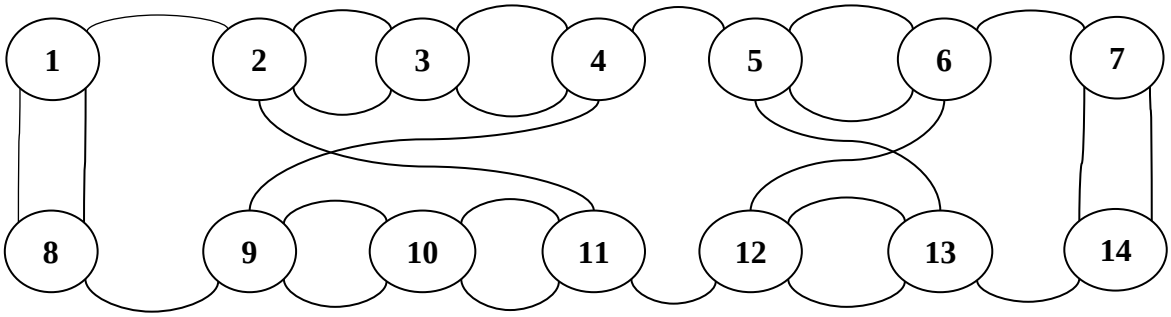
Структурата на системата може да се представи чрез графов модел $G=\{A, L\}$, при който множеството на върховете $\{A\} = \{a_1, a_2, ..., a_n\}$ съответства на множеството на елементите на системата (подсистемата, модула), а множеството на ребрата $\{L\} = \{l_1, l_2, ..., l_n\}$ – на връзките между елементите. Така задачата за декомпозиция на системата може да се приведе към задача за разделяне на графовия модел на подграфи по определен критерий.

Алгоритмите за декомпозиция, представени в първа глава, са приложени за системата, показана на *фиг.5.16*. Като определящ критерий е избран минимален брой външни връзки между отделените подсистеми.



Фиг.5.16. Структурна схема на система

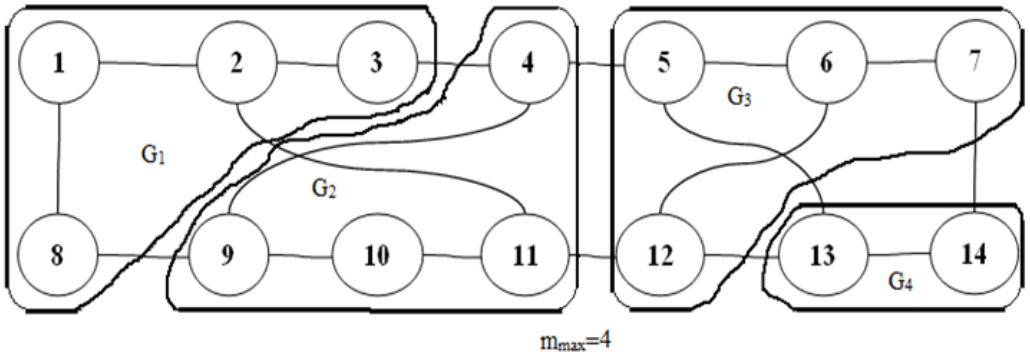
Построен е графовият модел с отчитане на паралелните ребра (връзки) *фиг.5.17*.



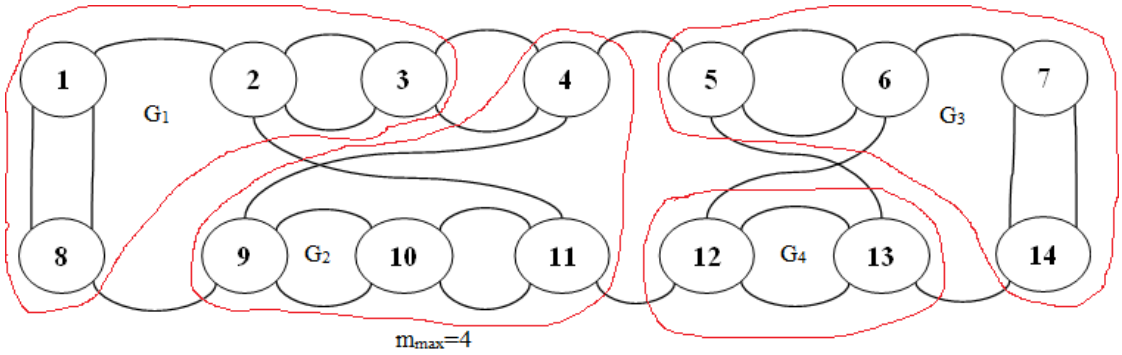
Фиг.5.17. Графов модел на системата на фиг. 5.16 с паралелни ребра

Разделянето на системата от *фиг.5.16* е реализирано с алгоритмите с добавяне и с отделяне, като те са приложени върху графовия модел.

На *фиг.5.19* се представено разделянето на графа без паралелни ребра, а на *фиг.5.23* – с паралелни ребра.



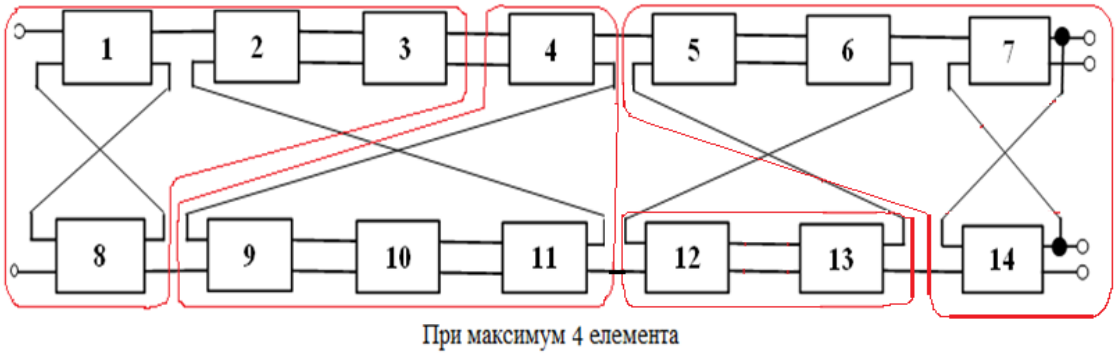
Фиг.5.19. Разделяне на графовия модел по алгоритъма с добавяне при $m_{max}=4$



Фиг.5.23. Разделяне на графовия модел от фиг.5.17 по алгоритъма с добавяне при $m_{max}=4$

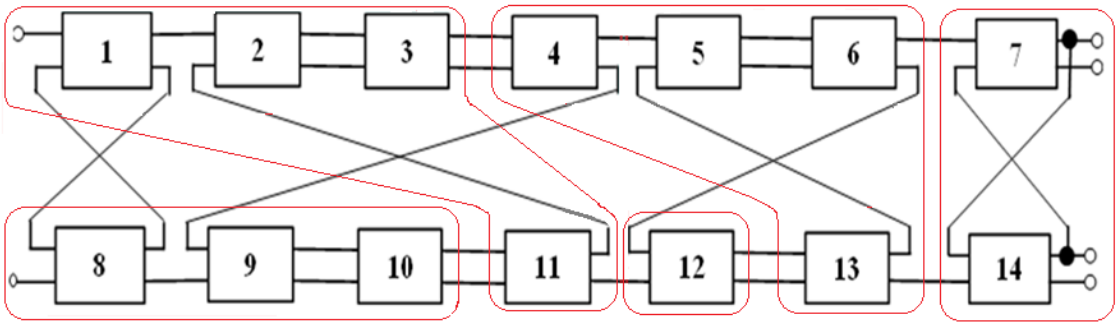
Вижда се фактът, че разликата е между подграфи G_3 и G_4. По поставения критерий оптималното разделяне е това на *фиг.5.23*.

Декомпозицията на графовия модел (*фиг.5.23*), приложена върху системата е показана на *фиг.5.28*.



Фиг.5.28. Разделяне на системата по алгоритъма с добавяне при максимум 4 елемента в подсистема (II-ри вариант)

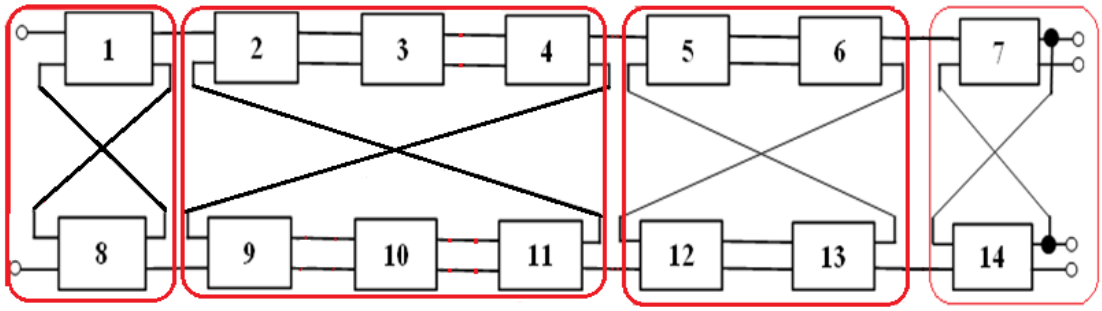
При разделянето на графовия модел на *фиг.5.17* по алгоритъма с отделяне за $m_{max}\geq 4$ се получават четири подграфа. След прилагането на тази декомпозиция върху системата се получава решението, показано на *фиг.5.30*.



При 4 или повече елемента

Фиг.5.30. Разделяне на системата по алгоритъма с отделяне

Най-доброто решение на задачата за декомпозиция се получава при максимум 6 елемента в една подсистема. Това е показано на *фиг.5.31*.



При максимум 6 елемента

Фиг.5.31. Разделяне на системата по най-добрия начин за максимум 6 елемента в подсистема

Подобно разделяне може да се получи по алгоритъма с добавяне, ако за следващ елемент не бъде избран, този с най-малък пореден номер.

Анализ на получените резултати.

Декомпозирането на схемата от *фиг.5.4* по алгоритъма с неприпокриване зависи от m_{max}. При $m_{max}>8$ разделянето е на две части, а при по-малки стойности – на три. Тези части невинаги са логически обособени модули, за разлика от алгоритъма с припокриване. От друга страна, на *фиг.5.13* се забелязва факта, че понякога алгоритъмът с неприпокриване разделя схемата на такива модули, което е за предпочитане. Той дава повече варианти за решение.

От получените резултати се установява, че при представянето на системата с графов модел е от значение отчитането на паралелните връзки при конструктивното проектиране. Това се вижда при сравнение на *фиг. 5.19* и *фиг.5.23*.

При решаването на задачата за декомпозиция на системи се установява, че алгоритъмът с добавяне е по-добър от алгоритъма с отделяне при системи с по-малки размери. От получените резултати се забелязва, че алгоритъмът

с добавяне разделя по оптимален начин системата при максимум 8 елемента. При намаляване на допустимия брой елементи в една подсистема се увеличава броят на външните връзки.

Изводи към Глава 5

- Алгоритъмът с припокриване разделя схемата на логически обособени модули, докато алгоритъмът с непripокриване невинаги прави това.
- По алгоритъма с непripокриване се получават повече варианти за разделянето на схемата.
- Алгоритмите с припокриване и непripокриване не са удачни за използване при конструктивно проектиране.
- При конструктивно проектиране е необходимо отчитането на паралелните връзки между елементите на системата, а при функционалното – не е.
- Алгоритъмът с отделяне невинаги е приложим за декомпозиция на системи.
- При системи с по-малки размери е по-удачно да се използва алгоритъмът с добавяне.

ЗАКЛЮЧЕНИЕ

На базата на осъществения обзор и анализа на състоянието по темата на дисертационния труд е направена класификация на алгоритмите за декомпозиция по различни критерии.

Изведени са математически изрази за оценка на сложността на изследваните алгоритми. Направена е оценка за сложност по време за всеки от тях.

Извършено е аналитично изследване на представените алгоритмите. Те са приложени при решаване на задачи за декомпозиция на графови модели с различен размер. На базата на получените резултати са обобщени изводи за алгоритмите и тяхната приложимост.

За целите на изследването са разработени специални програмни приложения. Чрез тях са определени броят на операциите и времето за изпълнение за всеки от изследваните алгоритми при декомпозиция на графови модели с различни размери.

Изследваните алгоритми са приложени при решаване на задачи от функционално и конструктивно проектиране на схеми и системи. Те могат да намерят приложение и при построяване на модели, изследване на обекти

с различни размери и сложност, проектиране на схеми, устройства и системи.

В. ПРИНОСИ НА ДИСЕРТАЦИОННИЯ ТРУД

1. Предложена е класификация на алгоритмите за декомпозиция.
2. Изведени са изрази за оценяване сложността на декомпозиционните алгоритми.
3. Определена е сложността на изследваните алгоритми.
4. Определен е броят на операциите и времето за изпълнение на декомпозиционните алгоритми в процеса на разделяне на графови модели с различни размери.
5. Разработени са програмни приложения, с които са изследвани алгоритмите за декомпозиция.
6. Алгоритмите с припокриване и неприпокриване са приложени при решаване на задачи от функционалното проектиране на схеми.
7. Алгоритмите с добавяне и отделяне са приложени при решаване на задачи от конструктивното топологическо проектиране на системи.

Г. СПИСЪК НА ПУБЛИКАЦИИТЕ ПО ДИСЕРТАЦИОННИЯ ТРУД

1. Динев М., Приложение на декомпозицията при автоматизирано проектиране, XXV Международна научна конференция „Мениджмънт и качество“, Ямбол, 11-12 май 2016, стр. 77-82, ISBN 978-619-160-679-5.
2. Dinev M., V. Kukenska, Matlab application for decomposition of graphs, XII International Conference Strategy of Quality in Industry and Education, Bulgaria, Varna, May 30 –June 2, 2016, pp 537 – 542, ISBN 978-966-2752-71-7.
3. Динев М., Структурна декомпозиция на графови модели, Международна научна конференция Унитех‘16 – Габрово, 18-19 ноември 2016, том II, стр. 215-218, ISSN 1313-230X.
4. Динев М., В. Кукенска, Програмно приложение за структурна декомпозиция, Международна научна конференция „Техника, Технологии, Образование“ – ICTTE 2017, Ямбол, 19-20 октомври 2017, стр. 190-197,ISSN 1314-9474
5. Динев М., В. Кукенска, Сравняване на алгоритмите за декомпозиция, V – научна конференция с международно участие „Компютърни науки и технологии“, Варна, 28-29 септември 2018, брой 2, стр. 29-33, ISSN 1312-3335

6. Динев М., В. Кукенска, П. Минев, Сложност на алгоритъм за декомпозиция, Международна научна конференция Унитех’18 – Габрово, 16-17 ноември 2018, том II, стр. 151-155, ISSN 1313-230X
7. Динев М., В. Кукенска, Класификация на алгоритмите за декомпозиция, Международна научна конференция Унитех’19 – Габрово, 15-16 ноември 2019, том II, стр. 127-130, ISSN 1313-230X
8. Динев М., Методика за оценяване на алгоритми за декомпозиция, Международна научна конференция Унитех‘20 – Габрово, 20-21 ноември 2020, том I, стр.384-389, ISSN 1313-230X
9. Кукенска В., М. Динев, П. Минев, И. Върбов. Приложение на принципа на декомпозиция при функционално и конструктивно проектиране, Международна научна конференция Унитех‘22 – Габрово,19-20 ноември 2022, том I,стр.318-323, ISSN 1313-230X

RESEARCH OF DECOMPOSITION ALGORITHMS

Matyo Stefanov Dinev

ABSTRACT

The dissertation presents the algorithms for decomposition representing the object of research, as well as their properties and characteristics. The criteria for their assessment have been formulated. A classification of decomposition algorithms is proposed.

The rules for calculating the complexity of an algorithm are defined. They are applied to the procedural steps of the decomposition algorithms. Through mathematical transformations, expressions determining the assessment of their complexity have been derived. The time complexity score for each of the studied algorithms was determined.

An analytical study of decomposition algorithms was performed. They have been applied in solving problems of splitting graph models of different sizes. Software applications have been developed for the purposes of the study. With them, a study of the presented algorithms for graph models with large dimensions was carried out. Based on these studies, results were obtained, which were used to evaluate and compare the studied algorithms.

Applications of decomposition algorithms in solving tasks from the field of functional and constructive design of schemes and systems using the principle of decomposition are presented. Based on the obtained results, conclusions were made about their applicability.

Keywords

decomposition algorithms, graph model, algorithm complexity, schemes, systems