МЕТОДИКА ЗА ОЦЕНЯВАНЕ НА АЛГОРИТМИ ЗА ДЕКОМПОЗИЦИЯ

Инж. Матьо Стефанов Динев[1]
[1]*Технически Университет - Габрово*

METHODOLOGY FOR EVALUATION OF DECOMPOSITION ALGORITHMS

Matyo Stefanov Dinev[1]
[1] *Technical University of Gabrovo*

Abstract

This paper deals with methodology for evaluating of the algorithms for decomposition of graph models. For each of them, evaluation of the approximate number of operations to be used is made. A research is conducted for the algorithms at different graph models.

Keywords: graph models; decomposition; algorithms.

ВЪВЕДЕНИЕ

Една от основните задачи, чието решение се налага при моделиране на обекти е задачата за декомпозиция. Тя е оптимизационна задача, при която обектите се разделят на функционално обособени модули. Структурата на всеки обект може да се представи чрез топологически граф. Задачата за декомпозиция може да се разглежда като задача за разделяне на графов модел.

При декомпозиция на графови модели се използват следните критерии: минимален брой външни връзки между подграфите; минимален брой подграфи; определен брой елементи в подграфите.

За разделянето на графа на подграфи се използват различни алгоритми за декомпозиция. За всеки от тях може да се определи сложността и да се направи оценка. Най – често за определянето ѝ се изчисляват брой операции за съответния алгоритъм.

В настоящия доклад се разглежда методика за оценяване на алгоритми за декомпозиция на графови модели. За всеки от тях е определен максималния брой използвани операции. Проведено е изследване на алгоритмите при различни графови модели.

МЕТОДИКА ЗА ОЦЕНЯВАНЕ

За методиката на оценяване на алгоритмите за декомпозиция са приети следните основни правила:

За една операция се приемат:

- Дефиниране и присвояване на стойност на променлива;
- Сравняване;
- Аритметична операция;
- Добавяне или вземане на елемент от масив (множество, граф и т.н.);
- Извикване на метод или функция.

За да се извърши оценяване на алгоритмите за декомпозиция се взема под внимание най – лошият случай (worst – case). Той дава информация за максималния брой използвани операции за изпълнението на съответния алгоритъм [1,2,3,6].

Чрез настоящата методика е направено оценяване на следните алгоритмите за декомпозиция:

- Алгоритъм с отделяне;
- Алгоритъм с добавяне;
- Алгоритъм с припокриване;
- Алгоритъм с неприпокриване.

Техните процедурни стъпки са описани в [4,5,6,7]. В [6] е извършено оценяване на алгоритъма с отделяне.

В следващия раздел е определена оценката на отделните стъпки за останалите три алгоритъма. Оценките на алгоритмите се отбелязват с буквата О и долен индекс отговаряш на първата буква от името на алгоритъма, както и номера на съответната стъпка. Представена е легенда за използваните означения при стъпките на алгоритъма.

ЛЕГЕНДА

G – граф за декомпозиране;
k – брой подграфи;
n – брой върхове за графа G
G_k – k-тия подграф за графа G;
α – вектор с елементи отговарящи на броя на граничещите ребра за всеки от върховете;
x_i – първи избран връх за съответния подграф;
I_k – множество съхраняващо избраните върхове;
P – множество съхраняващо теглата на избраните върхове от I_k.
J – двумерно множество съхраняващо външните съседи
M – множество съхраняващо теглата на външните съседи.
m – брой върхове за един подграф;
b_{ij} – брой външни връзки между *i-тия* и *j-тия* елемент
z – общи или външни върхове
$_{ij}$ – елемент от матрицата на съседство

ОЦЕНЯВАНЕ НА АЛГОРИТМИТЕ

ОЦЕНЯВАНЕ НА АЛГОРИТЪМА С ДОБАВЯНЕ

За определяне worst – case на този алгоритъм са разгледани следните два сценария:

I ви сценарий – когато броя на върховете в отделните подграфи е отново 1 (m=1);

II сценарий – когато броя на върховете в отделните подграфи е 2 (m=2).

В индекса на означението на оценката е прикрепен и номер за сценарий.

- *I-ви сценарий* - стъпките на алгоритъма ще се изпълняват *(n-1)* пъти, но стъпки 5 и 6 няма да се изпълнят нито веднъж. т.е. $O_{\partial I5}=O_{\partial I6}=0$.

<u>Стъпка 1:</u> В тази стъпка има 1 операция за присвояване на стойност на променлива, т.е. $O_{\partial I1}=1$.

<u>Стъпка 2:</u> За всеки n елемента на вектора α се сумират n събираеми. Операциите за събиране са с 1 по–малко от събираемите. Необходимо е и 1 операция за присвояване на крайния резултат към съответния елемент на α. Следователно за едно изпълнение на тази стъпка ще са необходими $(n\text{-}1+1)^2$ или n^2 операции. Тази стъпка се извършва за всеки подграф т.е. общо *(n-1)* пъти за сценарий I. За всяко следващо повторение на тази стъпка броя на върховете в графа G стават с един по – малко. От това следва, че за да се изчисли броят на операциите за тази стъпка, то трябва да се пресметне следната редица:

$$n^2+(n-1)^2+(n-2)^2+\ldots+(n-(n-2))^2$$

Преобразувайки я се получава следната формула (1).

$$\text{O}_{дI2}=\frac{n(n+1)(2n+1)}{6}-1 \qquad (1)$$

<u>Стъпка 3:</u> За да се определи $max(\alpha)$ трябва да се сравнят елементите от вектора α. Сравняванията са с 1 по – малко спрямо броя на елементите, следователно *(n-1)* сравнявания. За добавянето на елемента x_i към множеството I_k се използва 1 операция. Следователно необходимият брой операции за тази стъпка е n. Като се има в предвид сценарий I се повтаря *(n-1)* пъти, като за всяко повторение са необходими с една операция по – малко за сравняване (поради факта че върховете в G намалят с един). Получава се следната редица:

$$n+(n-1)+(n-2)+\ldots+(n-(n-2))$$

Пресмятайки я се получава следната формула (2):

$$\text{O}_{дI3}=\frac{n^2+n-2}{2} \qquad (2)$$

<u>Стъпка 4:</u> За определяне броят на елементите на множество съществуват определени функции. Поради тази причина това се брои за една операция. Освен това в тази стъпка има и 1 сравняване, така че операциите стават 2. Тази стъпка ще се повтаря *(n-1)* пъти за всеки подграф. Поради сценарий I условието ще е вярно още първия път и няма да има последващо извикване на тази стъпка за подграф. От което може да се направи извода че се получава следната формула:

$$O_{dI4} = 2(n-1) \quad (3)$$

При този сценарий стъпки 5 и 6 ще се прескачат, т.е. $O_{\partial I5}=0$ и $O_{\partial I6}=0$. За стъпка 7 има две операции (една за присвояване и една за събиране). Тя ще се изпълни за всеки един подграф, което означава *(n-1)* пъти. Следователно за $O_{\partial I7}$ се изчислява отново чрез формула (3), т.е. $O_{\partial I7}= O_{\partial I4}=2(n-1)$. Броят на операциите за стъпка 8 се изчислява по аналогичен начин т.е. $O_{\partial I8}= O_{\partial I7}$.

Общият брой операции за алгоритъма може да се определи от следната сума $O_{\partial I1}+O_{\partial I2}+O_{\partial I3}+O_{\partial I4}+O_{\partial I5}+O_{\partial I6}+O_{\partial I7}+O_{\partial I8}$, от която се получава формула (4):

$$O_{дIn} = \frac{n^3}{3} + n^2 + \frac{20}{3}n - 7 \quad (4)$$

- *II-ри сценарий*

При втория сценарий в зависимост от *n* дали е четно или не, се получават съответно *n/2* подграфи при четно *n*, или *(n+1)/2* подграфи при нечетно. Броят за изпълнения на отделните стъпки ще бъде равен на броя на подграфите, като за последния подграф стъпките не се изпълняват. За означението на оценката, към индекса е добавена буква „*н*“ или „*ч*“ означаваща съответно нечетни или четни стойности на *n*.

Стъпка 1: Стъпката е с константна оценка, така че броят операции отново е $O_{\partial II1}=1$.

Стъпка 2: Използвайки разсъжденията за стъпка 2 от алгоритъма с отделяне описан в [3], то броят на операциите може да се изчисли със следната редица:

$$\mathrm{n}^2 + (\mathrm{n}-2)^2 + (\mathrm{n}-4)^2 + \cdots$$

Опростявайки я се получава формула (5):

$$\mathrm{O}_{дII2} = \frac{n(n+1)(n+2)}{6} \quad (5)$$

Последният член на редицата при четни стойности за n е 2^2, а за нечетни е 1^2.Тъй като за последния подграф не се изпълняват стъпките, то следва от тази формула да се извади последния член на редицата. Така се получават формулите (6) и (7).

$$\mathrm{O}_{дII2ч} = \frac{n(n+1)(n+2)}{6} - 4 \quad (6)$$

$$\mathrm{O}_{дII2н} = \frac{n(n+1)(n+2)}{6} - 1 \quad (7)$$

Стъпка 3: За тази стъпка ще са необходими n на брой операции, който ще се повтарят толкова пъти колкото подграфи се получат. Имайки предвид сценарий II то следва че броя на върховете в *G* ще намаля с 2 за всяко следващо изпълнение на стъпките. От това се получава следната редица:

$$n + (\mathrm{n}-2) + (\mathrm{n}-4) + \cdots$$

Пресмятайки я и имайки в предвид, че последния член не се включва, то се получават формулите (8) и (9).

$$\mathrm{O}_{дII3ч} = \frac{(n+4)(n-2)}{4} \quad (8)$$

$$\mathrm{O}_{дII3н} = \frac{(n+3)(n-1)}{4} \quad (9)$$

Стъпка 4: Аналогично на стъпка 4 от предходния сценарий и при тази стъпка са необходими 2 операции. Имайки в предвид сценарий II тази стъпка ще се изпълни два пъти за всеки подграф. Следователно броя на операциите може да се изчисли чрез формули (10) и (11).

$$\mathrm{O}_{дII4ч} = 2.(n-2) \quad (10)$$

$$\mathrm{O}_{дII4н} = 2.(n-1) \quad (11)$$

Стъпка 5: В най – лошия вариант за тази стъпка е когато всеки връх в графа е свързан със всеки, т.е. съседите на върха x_i ще са *(n-1)*. Следователно броя на операциите за добавяне на съседи в множеството I_k е *(n-1)*. Стъпката ще се повтаря по два пъти за всеки подграф. Имайки в предвид и че броя на върховете в графа G ще намаля с 2 се получава следната редица.

$$2(n-1) + 2(\mathrm{n}-3) + 2(\mathrm{n}-7) + \cdots$$

Пресмятайки редицата се получават следните формули (12) и (13):

$$\mathrm{O}_{дII5ч} = \frac{n^2-4}{2} \quad (12)$$

$$\mathrm{O}_{дII5н} = \frac{n^2-9}{2} \quad (13)$$

Стъпка 6: В тази стъпка са необходими 4 операции за събиране, изваждане, умножение, и присвояване. Аналогично на предходните стъпки тези операции ще се изпълнят *(n-1)* пъти за този сценарий. Подобно на стъпка 3 са необходими и n операции за определянето и добавянето на минималния елемент b_{ij}. Стъпката ще се изпълни толкова пъти колкото подграфи се получат. За последния подграф се приемат останалите върхове и стъпката не се изпълнява. Така се получава следната редица:

{ 4(n-1) + n } + { 4(n-3) + (n-2) } + { 4(n-5) + (n-4) }+….

Пресмятайки я се получават следните две формули (14) и (15):

$$\mathrm{O}_{дII6ч} = \frac{(n-2)(5n+12)}{4} \quad (14)$$

$$\mathrm{O}_{дII6н} = \frac{(n-1)(5n+7)}{4} \quad (15)$$

Стъпка 7: Аналогично на сценарий I, стъпката има 2 операции който ще се изпълнят толкова пъти колкото са броя на подграфите без последния. Следователно след преобразуване на получаващия се израз се получава следните две формули (16) и (17):

$$O_{дII7ч} = (n-2) \quad (16)$$

$$O_{дII7н} = (n-1) \quad (17)$$

Стъпка 8: Аналогично на стъпка 7 се получава: $O_{дII8ч}=O_{дII7ч}$ и $O_{дII8н}=O_{дII7н}$.

Общият брой операции за алгоритъма, може да се определи от следните суми:
$O_{дII1}+ O_{дII2ч}+ O_{дII3ч}+ O_{дII4ч}+ O_{дII5ч}+ O_{дII6ч}+ O_{дII7ч}+ O_{дII8ч}$ – за четни стойности на n и $O_{дII1}+ O_{дII2н}+ O_{дII3н}+ O_{дII4н}+ O_{дII5н}+ O_{дII6н}+ O_{дII7н}+ O_{дII8н}$ – за нечетни стойности на n. Пресмятайки сумите получаваме следни формули (18) и (19), съответно за четни и нечетни стойности на n за сценарий II.

$$O_{дIIч} = \frac{1}{6}n^3 + \frac{5}{2}n^2 + \frac{16}{3}n - 21 \quad (18)$$

$$O_{дIIн} = \frac{1}{6}n^3 + \frac{5}{2}n^2 + \frac{16}{3}n - 11 \quad (19)$$

Сравнявайки формули (4),(18) и (19), като заместим n със различни стойности се вижда, че за сценарий I на алгоритъма ще са му необходими повече операции, въпреки че при него стъпките 5 и 6 ще бъдат пропуснати. При разглеждане на други сценарии (повечко върхове за един подграф), то броя на операциите значително ще намалее защото графа ще бъде разделен на по малък брой подграфи. Следователно би могло да се приеме че сценарий I е най лошия случай за този алгоритъм.

ОЦЕНЯВАНЕ НА АЛГОРИТЪМ С ПРИПОКРИВАНЕ

Стъпка 1: За избирането на върха с най – малка степен следва да се определи колко съседни върха има всеки връх в графа. За целта е необходимо от матрицата на съседство, отново да се изчисли вектора α и да се определи за кой връх α приема най – малка стойност. По аналогия на стъпка две от алгоритъма с отделяне описан в [6] са необходими: $\frac{n(n+1)(2n+1)}{6}$ операции за пресмятане на α и n операции за сравнение. От това следва че необходимия брой операции за стъпката е:

$$O_{п1} = \frac{n(n+1)(2n+1)}{6} + n \quad (20)$$

Стъпка 2: За тази стъпка са необходими толкова на брой операции колкото са съседните върхове. Стъпката ще се повтаря n на брой стъпки (за всеки избран връх в I_k). Най-лошия случай за тази стъпка е когато съседите са толкова на брой, колкото са всичките оставащи върхове извън множеството I_k. Следователно при първото изпълнение на стъпка 2 съседите ще са *(n-1)*, при второ *(n-2)*, при трето *(n-3)* и т.н. От тука следва, че за да изчислим броя операции за стъпката, то трябва да се пресметне следната редица: (n-1)+(n-2)+…..+3+2+1+0. След извършване на математическите действия се получава формула (20) чрез която може да се изчислят необходимия брой операции за тази стъпка.

$$O_{п2} = \frac{n(n-1)}{2} \quad (21)$$

Стъпка 3: За тази стъпка е необходимо да се пресметне теглото за избрания връх в I_k и то да се добави в множеството P. Това може да се направи с определена функция която по правилата се брои за 1 операция. За добавянето на теглото в P е необходима още една операция. Така за всяко изпълнение на тази стъпка ще са необходими по 2 операции. Следователно броя на операциите за стъпка 3 е $O_{п3}=2n$.

Стъпка 4: За проверката в тази стъпка е необходима 1 операция за сравнение. Стъпката ще се изпълни n пъти (за всеки избран връх в I_k). Следователно максималния брой операции е: $O_{п4}=n$

Стъпка 5: За определяне на теглото за всеки връх j в текущия ред на J е необходимо да се определят броя на външните съседи на върховете от подграфа образуван от множеството I_k и връх *j*. Това може да стане като се използва матрицата на съседство на графа G и се определят елементите на вектора α, като се имат в предвид само тези върхове които са външни за посочения подграф. При първото изпълнение на стъпката броя на тези върхове е (n-2), при второто изпълнение е (n-3) и т.н. Като се има в предвид че α се пресмята за всеки връх, като събираемите са броя на върховете то се получава следната редица:

$$(n-1)^2+(n-2)^2+(n-3)^2+\ldots+3^2+2^2+1^2$$

След като се определят теглата, то те трябва да се добавят в множество М. Броя на необходимите операции съответства на броя на външните върхове. Следователно за да се определят е необходимо да се пресметне следната редица:

(n-2)+(n-3)+(n-4)+….+3+2+1

Събирайки двете редици се получава че максималния брой операции за стъпка 5 се изчислява с формула (22).

$$O_{п5} = \frac{n(n-1)(2n-1)}{6} + \frac{(n-1)(n-2)}{2} \quad (22)$$

Стъпка 6: За намирането на позицията на най – малкото тегло от множеството M се извършват *(n-2)* на брой сравнявания. Ще е необходима и 1 операция за добавяне на елемент към множеството I_k. Тази стъпка ще се повтори *(n-1)* на брой пъти, колкото е броя на външните върхове. При всяко повторение броя на върховете ще намаля с един. Следователно броя на операциите може да се изчисли от следната редица:

{(n-2)+1}+{(n-3)+1}+….+{2+1}+{1+1}+{0+1}

Пресмятайки редицата се получава формула (23):

$$O_{п6} = \frac{n(n-1)}{2} \quad (23)$$

Стъпка 7: Подобно на стъпка 6 за намирането на позицията на най – малкото тегло ще са необходими m на брой операции. Тази стъпка ще се повтаря толкова на брой пъти колкото подграфи се получат, приблизително n/m. Следователно общия брой операции за тази стъпка, може да се пресметне от формула (24):

$$O_{п7} = \frac{nm}{m} = n \quad (24)$$

Стъпка 8: Тази стъпка образува k – тия подграф. В него се добавят избраните върхове от I_k. Това са върховете преди конкретната позиция. В най – лошия случай тази позиция би отговаряла на m. Подобно на предходната стъпка отново има n/m на брой повторения. Следователно $O_{п8}=O_{п7}=n$.

Стъпка 9: Аналогично на стъпка 8 се получава $O_{п9} = n$.

Стъпка 10: За изпълнението на стъпката, необходимия брой операции ще е 2 пъти по броя на общите върхове (z). Стъпката ще се изпълни толкова на брой пъти колкото подграфи се получат. Следователно максималния брой операции е: $O_{п10} = 2zn$.

Стъпка 11: Имайки в предвид правилото за брой операции при премахване на елемент от множество, то ще са необходими толкова операции колкото върхове ще се премахват, т.е. $O_{п11}=3n$.

Общият брой операции за алгоритъма може да се определи от следната сума $O_{п1}+O_{п2}+O_{п3}+O_{п4}+O_{п5}+O_{п6}+O_{п7}+O_{п8}+O_{п9}+O_{п10}+O_{п11}$, от която се получава следната формула:

$$O_{п} = \frac{2}{3}n^3 + \frac{3}{2}n^2 + \frac{47+2z}{6}n + 1 \quad (25)$$

ОЦЕНЯВАНЕ НА АЛГОРИТЪМА С НЕПРИПОКРИВАНЕ

Като се има в предвид най – лошия случай m=1, и че всеки връх е свързан с всеки, то алгоритъма, ще раздели съответния граф на един подграф. Това е така понеже ще се изпълнят само някои от стъпките на алгоритъма по веднъж. Върха в този граф ще бъде първия връх, а всички останали ще са отбелязани като външни. Следващите параграфи са оценени само стъпките които ще се изпълнят

Стъпка 1: Като се има в предвид най лошия случай и оценката от стъпка 1 на алгоритъма с припокриване се получава че $O_{н1}=O_{п1}$.

Стъпки 2: За тази стъпка са необходими толкова на брой операции колкото са и съседните върхове. За най лошия случай следва че $O_{н2}=(n-1)$.

Стъпка 3 и 4: Като се има в предвид анализа на оценките за стъпки 3 и 4 от алгоритъма с припокриване се стига до извода че $O_{н3}=2$ и $O_{н4}=1$.

Стъпки 8: За тази стъпка има само една стойност в множеството P. Следователно е необходима само една стъпка за извеждане на резултата, т.е. $O_{н8}=1$.

Стъпка 9: Въз основа на твърденията за стъпка 8 от алгоритъма с припокриване и това че тази стъпка няма да се повтаря, то следва че $O_{н9}=1$.

Стъпка 10: Броя на операциите в тази стъпка е равен на броя на външните върхове. За най лошия случай това е *(n-1)*. Т.е. $O_{н10}=(n-1)$.

Стъпка 11: За изчистване на множествата са необходими толкова на брой операции колкото върхове има във всяко от тях. Т.е. за множеството I_k е необходима *1* операция, за J – *(n-1)*,а за множеството P е необходима *1*. Следователно $O_{н11}=1+(n-1)+1=(n+1)$.

Останалите стъпки няма да се изпълнят и за тях не са необходими операции. Следователно общият брой операции за алгоритъма може да се определи от следната сума $O_{н1}+O_{н2}+O_{н3}+O_{н4}+O_{н8}+O_{н9}+O_{н10}+O_{н11}$ от която се получава формула (26):

$$O_{н} = \frac{1}{3}n^3 + \frac{1}{2}n^2 + \frac{25}{6}n + 4 \qquad (26).$$

РЕЗУЛТАТИ И ИЗВОДИ

С предложената методика са определени оценките за сложност на отделните алгоритмите за декомпозиция. Те са изследвани с различни графови модели, като е определен максималният брой операции за всеки от тях. Получените резултати са представени в табл. 1 и 2.

В табл. 1 е отразена оценката за алгоритмите с отделяне [6] и добавяне, а в табл. 2 - за алгоритмите с припокриване и неприпокриване. Тя е представена чрез максималния брой върхове за съответния алгоритъм. Оценката е отбелязана с буквата О и долен индекс отговарящ на първата буква от името на алгоритъма. С буквата n е отбелязан броя на върховете за съответния графов модел, за който е получена тази оценка.

Табл.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

Табл.2. *Оценка на алгоритмите с припокриване и неприпокриване*

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

За тези резултати могат да се направят следните изводи:

- С нарастване броят на върховете многократно нараства и броят на използваните операции
- За графови модели, при които теглото на върховете е по – голямо от максималния брой върхове за един подграф, то алгоритъма с добавяне е по – добър от алгоритъма с отделяне.
- За графови модели, при които теглото на върховете е по – голямо от максималния брой върхове за един подграф, то алгоритъма с неприпокриване е по – добър от алгоритъма с припокриване

ЗАКЛЮЧЕНИЕ

Разгледана е методика за оценяване на алгоритмите. Тя е приложена върху алгоритмите за декомпозиция. Представени са резултати от оценяване на алгоритмите. Получените резултати могат да се използват при избор на алгоритъм.

БЛАГОДАРНОСТИ

Настоящият документ е изготвен с финансовата помощ на договор № 2006Е за провеждане на научни изследвания по проект на тема: „IoT-базирана услуга за персонализирано доставяне на съдържание и автоматично профилиране на клиентите“ към Технически университет – Габрово.

REFERENCE

[1]. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest., Introduction to Algorithms. Cambridge, MA: MIT Press, 1990.

[2]. Nakov P., P. Dobrikov, Programming = ++Algorithms, TopTeam Co., Sofia, 2002.

[3]. Soltys M., An Introduction to the Analysis of Algorithms 2nd Edition, New Jersey: World Scientific, 2012.

[4]. Dinev M., V. Kukenska, MATLAB application for decomposition of graphs, XII International Conference Strategy of Quality in Industry and Education, Varna, Bulgaria, May 30 – June 02 2016, pp.537-542, ISBN 978-966-2752-71-7.

[5]. Dinev M., Structural Decomposition of graphs models, International Scientific Conference Unitech’2016 – Gabrovo, 18 – 19 November 2016, Volume II, pp. 215-218, ISSN 1313-230X.

[6]. Dinev M., V. Kukenska, P. Minev, Complexity of decomposition algorithm, International Scientific Conference Unitech’2018 – Gabrovo, 16 – 17 November 2018, Volume II, pp. 151-155, ISSN 1313-230X.

[7]. Dinev M., V. Kukenska, Clasification of decomposition algorithm, International Scientific Conference Unitech’2019 – Gabrovo, 15 – 16 November 2019, Volume II, pp. 127-130, ISSN 1313-230X.