УДК 681.3

АЛГОРИТМ РОЗКЛАДАННЯ ГРАФІВ ЗА ДОПОМОГОЮ ЇХНІХ КІСТЯКІВ

Т. Гришанович

Київський національний університет імені Т.Г.Шевченка
просп. академіка Глушкова, 1, 03187 Київ, Україна
e-mail: grtanja@rambler.ru

Розглянуто питання розкладання (розфарбування, декомпозиції) графів. Зокрема, описано метод та алгоритм розкладання графа за допомогою його кістяків. Наведено приклади роботи такого алгоритму для неорієнтованих ненавантаженого графів різної розмірності, а також приклад програмної реалізації конкретною мовою програмування. Окреслено головні проблеми, що виникли під час побудови нормальних кістяків графів.

Ключові слова: розкладання графів, алгоритм розкладання, алгоритм декомпозиції.

У разі розкладання графів звертаються до двох головних задач: розфарбування (розкладання) вершин та ребер. Розкладання ребер графа полягає у приписуванні цим ребрам таких кольорів, що жодні два суміжні ребра не мають однакового кольору. Такий розподіл кольорів називають власним розфарбуванням ребер. Аналогічно означають і розкладання вершин графа, коли всі його суміжні вершини мають різні кольори. У цьому випадку говорять про власне розфарбування вершин графа. Реберно-хроматичним числом графа називають найменшу кількість кольорів, необхідну для власного розфарбування ребер графа. Вершинно-хроматичне число, або хроматичний індекс, означають як найменшу кількість кольорів, необхідну для власного розфарбування вершин графа.

Проблема розкладання вершин полягає у відшуканні хроматичного числа заданого графа [1]. Основною проблемою в разі розв'язування задач розфарбування вершин графів є те, що компоненти розбиттів або декомпозицій у напрямах теорії графів мають бути зв'язними, тоді як компоненти оптимальних розкладань, як звичайно, виявляються незв'язними. Це значно утруднює застосування традиційних методів теорії графів для розв'язування задач розкладання [2].

Питанням розбиття графів присвячені праці [2–4]. У [5] запропоновано такі способи розкладання графів: індуктивний, розкладання за незалежними множинами, розкладання за кістяками, розкладання за узагальненими циклами та променями. Наша мета – побудова алгоритму розкладання графів за допомогою їхніх кістяків на підставі запропонованого методу та його реалізації.

Метод розкладання. Для того, щоб описати метод розкладання графів за допомогою їхніх кістяків, означимо деякі головні поняття та сформулюємо теорему.

Означення. Дерево з виділеним коренем x назвемо x-кореневим.

Означення. Визначимо частковий порядок $<$ на множині V вершин графа $Gr(V, E)$ за таким правилом: $y < z$ тоді й тільки тоді, коли найкоротший шлях від x до z проходить через вершину y.

Означення. Кістяк $Tr(V, E')$ графа $Gr(V, E)$ називають нормальним x-кореневим**,** якщо будь-які дві суміжні вершини графа $Gr(V, E)$ є порівняними в частковому порядку на V**,** визначеному x-кореневим деревом $Tr(V, E)$.

Теорема. Нехай $Gr(V, E)$ – зв'язний граф, r – натуральне число, $|V| \le r$. Тоді існує r-розфарбування χ: $V \to \{1, 2, \dots r\}$ таке, що кожна куля $B(v, k)$, $v \in V$, $k \in \{1, 2, \dots r-1\}$ містить принаймні $k+1$ вершину, що пофарбовані різними кольорами [5].

Нехай маємо граф, що є деревом Tr з коренем x, причому відстань від будь-якої кінцевої вершини (якщо такі є) до кореня не менше r-1. Визначимо розфарбування χ множини вершин дерева кольорами $\{0,1\dots r\text{-}1\}$ за таким правилом:

$$\chi(v) = d(v, x) \bmod r \,. \qquad (1)$$

Зрозуміло, що кожна куля радіусом k, $k \in \{0,1,\dots r\text{-}1\}$ містить принаймні $k+1$ вершин різних кольорів. Отже, якщо для графа Gr(V, E) з наведеної вище теореми ми побудували кістяк Tr з потрібною властивістю, то автоматично одержуємо розфарбування χ.

Очевидно, що не кожен зв'язний граф $Gr(V, E)$, що має $\ge r$ вершин, має також і потрібний кістяк. Ми опишемо конструкцію такого кістяка за додаткової умови: граф $Gr(V, E)$ скінченний і $\rho(v) \ge r\text{-}1$ для всіх вершин $v \in V$. Цю конструкцію і будемо використовувати надалі.

Доведемо, що кожен скінченний зв'язний граф має нормальний x-кореневий кістяк для будь-якої вершини $x \in V$. Почнемо з довільного x-кореневого кістяка $Tr(V, E_0)$.

$$\delta(Tr(V, E_0)) = \sum_{v \in V} d(x, v) \,,$$

де d – відстань від вершини v до кореня в графі $Tr(V, E)$. Якщо кістяк $Tr(V, E_0)$ нормальний, то побудову закінчено.

Припустимо, що $Tr(V, E_0)$ не є нормальним x-кореневим кістяком. Тоді в графі $Gr(V, E)$ знайдуться дві суміжні вершини v, u, що є непорівнянними в частковому порядку, визначеному деревом $Tr(V, E_0)$. Нехай для визначеності $d(u, x) \ge d(v, x)$. Зрозуміло, що $x \ne v$, $x \ne u$. Виберемо вершину v' таку, що $(v, v') \in E_0$ та $d(x, v') = d(x, v)\text{-}1$. Вилучимо ребро (v, v') та додамо нове ребро (u, v). Отримаємо новий x-кореневий кістяк $Tr(V, E_1)$. Зазначимо, що

$$\delta(Tr(V, E_1)) > \delta(Tr(V, E_0)) \,.$$

Якщо кістяк $\delta(Tr(V, E_1))$ графа $Gr(V, E)$ не є нормальним, то повторимо для нього цю ж процедуру, оскільки

$$\delta(Tr(V, E_0)) < (n-1)d,$$

де d – максимальна відстань від між вершинами графа $Gr(V, E)$. Через скінченнк кількість кроків ми отримаємо нормальний x-кореневий кістяк. До цього дерева і застосуємо розфарбування (1).

Очевидно, що відстань від будь-якої кінцевої вершини v дерева $Tr(V, E')$ до кореня не менша від r-1, адже всі суміжні з v вершини графа $Gr(V, E)$ лежать на найкоротшому шляху від v до x. Отже, розфарбування χ за-

довольняє наведену вище теорему [5]. Описаний спосіб називають розкладанням графів за допомогою їхніх кістяків.

Алгоритм розкладання. Нехай $Gr(V, U)$ – неорієнтований зв'язний ненавантажений граф. Алгоритм побудови розбиття графа за допомогою кістяків матиме такий вигляд:

1. Побудувати кістяк графа. У підсумку отримаємо кістяк $Tr(V, U')$.
2. Виділити кореневу вершину. Визначити частковий порядок $\leq$ на множині V вершин графа за таким правилом: $v_i \leq v_j$ тоді й тільки тоді, коли найкоротший шлях від вершини v_j до кореневої вершини проходить через v_i.
3. Перевірити чи отриманий кістяк є нормальним, тобто чи всі суміжні вершини дерева є порівнюваними в частковому порядку, визначеному кореневою вершиною.
4. Якщо кістяк нормальний, то застосувати до графа розфарбування

$$\chi(v_i)=d(v_i, x) \bmod r,$$

де $d(v_i, x)$ – мінімальна відстань від кореневої вершини x до вершини v_i, r – кількість кольорів, якими можна розфарбувати граф; $r \leq \rho(v)+1$, де $\rho(v)$ – максимальний серед степенів вершин графа [1].
5. Якщо кістяк не є нормальними, то знайдуться дві вершини v_i та v_j, що не порівнювані в частковому порядку, визначеному деревом $Tr(V, U')$: визначити відстані $d(v_i, x)$ та $d(v_j, x)$; якщо $d(v_i, x) \geq d(v_j, x)$, тоді вибрати вершину $v_к$ таку, що $(v_j, v_к)$ належить U' та $d(v_к, x) = d(v_j, x)-1$;.видалити ребро $(v_j, v_к)$ та додати ребро (v_j, v_i); якщо $d(v_i, x) \leq d(v_j, x)$, то повторити описану вище операцію для вершини v_i; отримаємо кістяк $Tr(V, U'')$.
6. Перейти до кроку 3.

Зазначимоти, що для довільного скінченного графа можна отримати нормальний x-кореневий кістяк за скінченну кількість кроків. Так само і кожен зліченний граф має нормальний кореневий кістяк. Доведення цього твердження для локально-скінченних зв'язних графів наведено у [5].

Приклад. Для зменшення кількості ітерацій у разі відшукання нормального кістяка графа виберемо кількість вершин 3:n=3. Граф задамо матрицею суміжності ($n \times n$), де n – кількість вершин графа. Якщо вершини v_i та v_j не з′єднані між собою ребром, то на перетині i-го та j-го рядка розміщене число 1 000, у протилежному випадку, якщо ребро (v_i, v_j) існує, – число 1. Якщо ж $i=j$, то на перетині i-го та j-го рядка буде число 0.

Нехай маємо граф $Gr(V, U)$: n=3. Його матриця суміжності має такий вигляд:

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}$$

1. Побудуємо кістяк графа $Gr(V, U)$. Для цього використаємо алгоритм Пріма–Краскала.[2]. Отримаємо кістяк $Tr(V, U')$. Він матиме такий вигляд:

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1000 \\ 1 & 1000 & 0 \end{pmatrix}$$

2. Виділимо кореневу вершину. Кореневою є та вершина, максимальна відстань від якої до всіх інших вершин графа дорівнює радіусу цього дерева [6]. Для визначення відстані між вершинами дерева використаємо алгоритм Дейкстри [7]. Для нашого дерева кореневою є вершина v_1 .

Визначимо частковий порядок $\leq$ на множині V вершин графа за таким правилом: $v_i \leq v_j$ тоді і тільки тоді, коли найкоротший шлях від вершини v_j до кореневої вершини проходить через v_i.

3. Перевіримо, чи отриманий кістяк $Tr(V, U')$ є нормальним. Розпочнемо з вершини v_2. Вершини v_2 та v_3 не порівнювані за частковим порядком, визначеним кореневою вершиною, оскільки найкоротший шлях від вершини v_3 не проходить через вершину v_2. Отже, триманий кістяк не є нормальним.

4. Визначимо відстань $d(v_2, v_1)$ та $d(v_3, v_1)$: $d(v_2, v_1)=d(v_3, v_1)=1$. Виберемо вершину v_k таку, що $d(v_k, v_1) = d(v_3, v_1)-1$. У нашому випадку цією вершиною буде v_1. Видалимо ребро (v_3, v_1) та додаємо ребро (v_3, v_2). Отримано кістяк $Tr'(V, U'')$.

5. Перевіримо, чи кістяк $Tr'(V, U'')$ нормальний. Вершини v_2 та v_3 порівнювані за частковим порядком $\leq$, визначеним кореневою вершиною. Отже, отриманий кістяк нормальний.

6. Застосуємо до графа $Tr'(V, U'')$ розфарбування (1). Оскільки максимальний степінь графа $Gr(V, U)$ дорівнює 2, то його вершини можна розфарбувати максимум $r \leq 3$ кольорами [6]:

1) $r=2$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=0$;
2) $r=3$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$.

Отже, ми отримали розкладання графа $Gr(V, U)$ за допомогою його кістяків.

Розроблений алгоритм апробовано на низці тестових задач, частину з яких наведено нижче.

1. $N=4$

$$\begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Нормальний кістяк:

$$\begin{pmatrix} 0 & 1 & 1000 & 1000 \\ 1 & 0 & 1 & 1000 \\ 1000 & 1 & 0 & 1 \\ 1000 & 1000 & 1 & 0 \end{pmatrix}$$

Розфарбування графа:

двома кольорами $r=\{0, 1\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=0$, $\chi(v_4)=1$;
трьома кольорами $r=\{0, 1, 2\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=0$;
чотирма кольорами $r=\{0, 1, 2, 3\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=3$.

2. N=5

$$\begin{pmatrix} 0 & 1 & 1 & 1000 & 1 \\ 1 & 0 & 1 & 1000 & 1000 \\ 1 & 1 & 0 & 1 & 1000 \\ 1000 & 1000 & 1 & 0 & 1 \\ 1 & 1000 & 1000 & 1 & 0 \end{pmatrix}$$

Нормальний кістяк:

$$\begin{pmatrix} 0 & 1 & 1000 & 1000 & 1000 \\ 1 & 0 & 1 & 1000 & 1000 \\ 1000 & 1 & 0 & 1 & 1000 \\ 1000 & 1000 & 1 & 0 & 1 \\ 1000 & 1000 & 1000 & 1 & 0 \end{pmatrix}$$

Розфарбування графа:

двома кольорами $r=\{0, 1\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=0$, $\chi(v_4)=1$, $\chi(v_5)=0$;
трьома кольорами $r=\{0, 1, 2\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=0$, $\chi(v_5)=1$;
чотирма кольорами $r=\{0, 1, 2, 3\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=3$, $\chi(v_5)=0$.

3. N=6

$$\begin{pmatrix} 0 & 1 & 1000 & 1000 & 1000 & 1000 \\ 1 & 0 & 1 & 1000 & 1 & 1000 \\ 1000 & 1 & 0 & 1 & 1000 & 1 \\ 1000 & 1000 & 1 & 0 & 1 & 1000 \\ 1000 & 1 & 1000 & 1 & 0 & 1000 \\ 1000 & 1000 & 1 & 1000 & 1000 & 0 \end{pmatrix}$$

Нормальний кістяк:

$$\begin{pmatrix} 0 & 1 & 1000 & 1000 & 1000 & 1000 \\ 1 & 0 & 1 & 1000 & 1000 & 1000 \\ 1000 & 1 & 0 & 1 & 1000 & 1 \\ 1000 & 1000 & 1 & 0 & 1 & 1000 \\ 1000 & 1000 & 1000 & 1 & 0 & 1000 \\ 1000 & 1000 & 1 & 1000 & 1000 & 0 \end{pmatrix}$$

Розфарбування графа:

двома кольорами $r=\{0, 1\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=0$, $\chi(v_4)=1$, $\chi(v_5)=0$, $\chi(v_6)=1$;

трьома кольорами $r=\{0, 1, 2\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=0$, $\chi(v_5)=2$, $\chi(v_6)=0$;

чотирма кольорами $r=\{0, 1, 2, 3\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=3$, $\chi(v_5)=2$, $\chi(v_6)=3$.

3. $N=7$

$$\begin{pmatrix} 0 & 1 & 1 & 1000 & 1000 & 1000 & 1000 \\ 1 & 0 & 1 & 1000 & 1000 & 1000 & 1000 \\ 1 & 1 & 0 & 1 & 1000 & 1000 & 1000 \\ 1000 & 1000 & 1 & 0 & 1 & 1000 & 1 \\ 1000 & 1000 & 1000 & 1 & 0 & 1 & 1000 \\ 1000 & 1000 & 1000 & 1000 & 1 & 0 & 1 \\ 1000 & 1000 & 1000 & 1 & 1000 & 1 & 0 \end{pmatrix}$$

Нормальний кістяк:

$$\begin{pmatrix} 0 & 1 & 1000 & 1000 & 1000 & 1000 & 1000 \\ 1 & 0 & 1 & 1000 & 1000 & 1000 & 1000 \\ 1000 & 1 & 0 & 1 & 1000 & 1000 & 1000 \\ 1000 & 1000 & 1 & 0 & 1 & 1000 & 1000 \\ 1000 & 1000 & 1000 & 1 & 0 & 1 & 1000 \\ 1000 & 1000 & 1000 & 1000 & 1 & 0 & 1 \\ 1000 & 1000 & 1000 & 1000 & 1000 & 1 & 0 \end{pmatrix}$$

Розфарбування графа:

двома кольорами $r=\{0, 1\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=0$, $\chi(v_4)=1$, $\chi(v_5)=0$, $\chi(v_6)=1$, $\chi(v_7)=0$;

трьома кольорами $r=\{0, 1, 2\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=0$, $\chi(v_5)=1$, $\chi(v_6)=2$, $\chi(v_7)=0$;

чотирма кольорами $r=\{0, 1, 2, 3\}$: $\chi(v_1)=0$, $\chi(v_2)=1$, $\chi(v_3)=2$, $\chi(v_4)=3$, $\chi(v_5)=0$, $\chi(v_6)=1$, $\chi(v_7)=2$.

Програмна реалізація. На вході програма отримує кількість вершин графа (n) та матрицю суміжності – matrix. Якщо вершини з номерами i та j не з'єднані між собою ребром, то на перетині i-го та j-го рядка розміщене число 1 000, у протилежному випадку – число 1. Якщо $i=j$, то на перетині i-го та j-го рядка розміщене число 0 [7].

Основна частина програми виконує безпосередньо відшукання одного з кістяків графа (початкового), кореневої вершини та кореневих нормальних кістяків. Коли такий кістяк знайдено, відбувається побудова розфарбування графа:

```
Const
n; // максимальна кількість вершин графа
Type
graph=array [1..n, 1..n] of integer; // структура матриці суміжності
```

```
var
matrix: graph; // матриця суміжності
ks:  graph; // кістяк графа
x: integer; // коренева вершина
nn: integer; // фактична кількість вершин графа
max: integer// максимальний степінь вершин графа
norm: array [1..20] of  record // масив для перевірки того, чи є кістяк нормальним
  a11, a12: integer; // номери вершин – закінчень ребра
  b11: boolean; // чи перебувають вершини a11 та a12 у відношенні часткового порядку
   end;
kistjak: boolean; // змінна, що набуває значення True в тому випаду, коли масив norm не містить жодного елемента b11=false, та значення false – в усіх інших випадках
hi: array [1..n] of integer; // масив, у якому зберізається розфарбування графа
BEGIN
// ініціалізація масиву matrix, визначення максимального степеня графа
 // відшукання мінімального кістяка графа (алгоритм Пріма-Краскала)
// відшукання кореневої вершини
  kistjak:=false;
 repeat
 BEGIN
{ побудова допоміжного масиву norm для визначення того, чи кістяк нормальний}
                dejkstr(nn, ks, x, i, l1, k1, path); // процедура, що реалізує алгоритм Дейкстри відшукання мінімальної відстані між вершинами; x, i – номери вершин, відстань між якими обчислюється; l1 – допоміжна змінна; k1 – відстань між вершинами x та i; path – масив, що містить шлях  від x до i.
                dejkstr(nn, ks, x, j, l2, k2, path1);
                    if path1[p]=i then
                    norm[h].b11:=true else  norm[h].b11:=false;
          {ітерація виправлення кістяка на нормальний; даний блок команд повторюється доки масив norm не міститиме жодного елемента b11=false }
if norm[i].b11 = false then
      dejkstr(nn, ks, x, norm[i].a11, l1, k1, path);
      dejkstr(nn, ks, x, norm[i].a12, l2, k2, path1);
      if k1<=K2 then
         j:=norm[i].a12;
            dejkstr(nn, ks, x, p, l1, k3, path);
            if (matrix[p, j]<>0) and (matrix[p, j]<>1000) and (ks[p, j]<>0) and(k3<=k1-1) then
// видаляємо ребро (p, j)
                    ks[norm[i].a11, norm[i].a12]:=1;
                    ks[norm[i].a12, norm[i].a11]:=1;
                    norm[i].b11:=true;
                  else
                j:=norm[i].a12;
                dejkstr(nn, ks, x, p, l1, k3, path);
```

```
                if (matrix[p, j]<>0) and (matrix[p, j]<>1000) and (ks[p, j]<>0)
                and(k3<=k1-1) then
        // видаляємо ребро (p, j)
                    ks[norm[i].a11, norm[i].a12]:=1; // додаємо нове ребро
                    ks[norm[i].a12, norm[i].a11]:=1;
                    norm[i].b11:=true;
                {перевіримо, чи масив norm містить значення false}
                if norm[i].b11=false then kistjak:=false;
  until kistjak=false;
    {розфарбування отриманого нормального кореневого кістяка усіма можливими кольорами}
  hi[i]:= trunc(k1)mod a;
END.
```

На виході програми отримаємо нормальний кореневий кістяк, який також задають матрицею суміжності, та усі можливі варіанти розфарбування графа.

Отже, розроблено та реалізовано алгоритм розбиття графа за допомогою його кістяків. Підтвердження ефективності алгоритму продемонстровано низкою обчислюваних експериментів з використанням розробленого програмного забезпечення.

Головна проблема, що виникла під час розробки алгоритму, зокрема в разі практичної реалізації, полягає у визначенні кількості ітерацій побудови дерев для відшукання нормального кореневого кістяка графа. У випадку програмної реалізації алгоритму питання вирішене так. Використовують масив записів norm, що має таку структуру: $a11$, $a12$ – елементи цілого типу, що відповідають номерам вершин, закінченням ребра, $b11$ – величина логічного типу, що набуває значення ІСТИНА в тому випадку, коли вершини, номери яких містяться у значеннях величин $a11$ та $a12$, порівнювані у частковому порядку, заданому кореневою вершиною, та значення ХИБНО – у протилежному випадку. Так після побудови кожного кістяка графа $Gr(V, E)$ будують масив norm. У тому випадку коли масив містить хоча б один елемент зі значенням змінної $b11$ ХИБНО, відбувається побудова нового кістяка. У протилежному випадку будують розфарбування графа. Однак незважаючи на те, що описана вище проблема ефективно вирішена під час розробки програмного забезпечення, питання теоретичного обґрунтування кількості ітерацій у разі побудови нормального кореневого кістяка графа є відкритим.

Подання графа за допомогою матриці суміжності не є єдиним способом відображення графів у ЕОМ. До них також належать матриці інцидентності, списки суміжності, масиви дуг та двійкове зображення [8]. Тому надалі плануємо оцінити однин з видів складності алгоритму, наприклад, часову, залежно від способу зображення графа.

---

1. *Новиков Ф.А.* Дискретная математика для программистов. СПб.: Питер, 2003. 301 с.
2. *Протасов І.В., Протасова К.Д.* Розкладність графів: Навч. посібник. К.: Київс. ун–т, 2003. 73 с.
3. *Protasov I., Banakh T.* Ball Structures and Colorings of Graphs and Groups // Mat. Stud. Monogr. Ser. VNTL. Lviv, 2003. Vol. 11. 147 p.
4. *DiesteI R. Graph Decomposition.* A Study in Infinite Graphs. Oxford: Clarendon Press, 1990. 350 p.

5. *Протасова К.Д.* Розкладання графів: Дис. канд. фіз.-мат. наук. К., 2006.
6. *Горбатов В.А.* Фундаментальные основы дискретной математики. М.: Наука-Физматлит, 2000. 208 с.
7. *Касьянов В.Н., Евстигнеев В.А.* Графы в программировании: обработка, визуализация и применение. СПб.: БХВ-Петербург, 2003. 104 с.
8. *Асельдеров З.М.* Представление и восстановление графов. К.: Наук. думка, 1991.

GRAPHS DECOMPOSITION ALGORITHM BY MEANS OF ITS CARCASS

T. Grishanovich

Faculty of Cybernetics, Taras Shevchenko National University of Kyiv
1 Academician Glushkov Ave., UA–03187 Kyiv, Ukraine
grtanja@rambler.ru

This study is devoted to the problems of decomposition (‘partition’ or ‘coloring’) of graphs. We describe a technique for partitioning by means of the graph’s carcass, along with the corresponding algorithm. Besides, an example of work of this algorithm is described in a more detail for the case of non-oriented unloaded graphs. We also give an example of program realization of the above algorithm using a specific programming language. Basic problems that appear when constructing normal trees of the graphs are described.

Key words: graph partition, coloring algorithm, graph decomposition algorithm.

АЛГОРИТМ ДЕКОМПОЗИЦИИ ГРАФОВ С ПОМОЩЬЮ ИХ ОСТОВОВ

Т. Гришанович

Киевский национальный университет имени Т.Г. Шевченко
факультет кибернетики
просп. академика Глушкова, 1, 03187 Киев, Украина
grtanja@rambler.ru

Работа посвящена вопросам декомпозиции (раскрашивания, разбиения) графов. В ней описаны метод и алгоритм раскрашивания графа с помощью его остовов. Кроме того, наведен пример работы такого алгоритма для неориентированных графов разной размерности. Представлен пример программной реализации алгоритма на конкретном языке программирования. Очерчены основные проблемы, возникающие при построении нормальных остовов графов.

Ключевые слова: раскраска графов, алгоритм раскраски, алгоритм декомпозиции.

Стаття надійшла до редколегії 29.04.2009
Прийнята до друку 30.06.2009