Графи. Дървета. Обхождане на графи.

Дефиниция: Нека $V = \{v_1, v_2, v_3, \ldots, v_n\}$ е крайно множество, елементите на което наричаме върхове, а $E = \{e_1, e_2, e_3, \ldots e_m\}$ е крайно множество, елементите на което наричаме ребра. Функцията $f_G : E \to V \times V$, съпоставяща на всяко ребро наредена двойка от върхове, наричаме *краен ориентиран мултиграф*. Казваме още, че е зададен краен ориентиран мултиграф G с върхове V и ребра E и свързваща функция f_G и го означаваме с G(V, E, f_G).

Графите представяме в ранината, като всеки връх $v_i \in V$ представяме с точка, а реброто $e \in E$, такова че $f_G(e) = (v_i, v_j)$ изобразяваме с линия, започваща във v_i и завършваща със срелка във v_j (за да се посочи кой е вторият елемент на наредената двойка). Ребрата $f_G(e) = (v_i, v_i)$ наричаме *примки*. Името на реброто записваме до линията.

***Дефиниция*:** Нека G(V, E, f_G) е краен ориентиран мултиграф и функцията f_G е еднозначна, т.е. f(e_i) ≠ f(e_j), i ≠ j. Тогава G(V, E, f_G) наричаме *краен ориентиран граф.*

***Дефиниция*:** Нека G(V, E) е краен ориентиран граф, такъв че релацията $E \subseteq V \times V$ е рефлексивна и симетрична. В такъв случай G(V, E) наричаме *краен неориентиран граф* или просто *граф*.

Забележка: Крайният неориентиран граф G(V, E) можем да превърнем в *краен неориентиран мултиграф*, ако позволим повече от едно неориентирано ребро да свързва два върха от V, т.е. ако вместо множество $E \subseteq V \times V$ вземем мултимножество от елементите на $V \times V$.

Забележка: Връзката между крайните неориентитани графи и крайните неориентирани мултиграфи е както между конфигурациите без наредба и без повторение (множествата) и конфигурациите без наредба с повторение (мултимножества).

***Дефиниция*:** Върховете $v_i, v_j \in V$ на графа G(V, E) наричаме *съседни*, ако (v_i, v_j) ∈ *E*. Ако графът е ориентиран казваме че v_i е *баща* на v_j, а v_j е *син* на v_i. Казваме че v_i и v_j са *краища* на реброто (v_i, v_j). Броят d(v_i) на ребрата в неориентирания граф G(V, E), на които $v_i \in V$ е край наричаме *степен* на v_i. Броят $d^-(v_i)$ на ребрата в ориентирания граф G(V, E), които започват от $v_i \in$ V, наричаме *полустепен на изхода* на v_i, а броят на $d^+(v_i)$ на ребрата, които

завършват във $v_i \in V$, наричаме *полустепен на входа* на v_i.Ако d(v_i) = 0 (v_i няма съседи), то ще наричаме v_i *изолиран.*

***Дефиниция*:** Нека G(V, E, f_G) е краен ориентиран мултиграф, $V' \subseteq V$. Нека $E' \subseteq E$ се състои от тези ребра, краищата на които са във V', а f'_G (e) = f_G (e), $\forall$ e$\in E'$ (f'_G е рестрикцията на f_G върху E'). Тогава крайният ореинтиран мултиграф $G'(V', E', f'_G)$ наричаме *подмултиграф на G*. Ако G е краен ориентиран или неориентиран граф, то $G'(V', E')$ наричаме *подграф* на G. И в двата случая казваме, че G' е *индуциран* от V' подграф на G.

***Дефиниция*:** Матрицата M = $||a_{ij}||$ с размери $|V| \times |V|$ наричаме *матрица на съседства* на крайният ориентиран мултиграф G(V, E, f_G), ако $\forall\, v_i, v_j \in V$ е в сила $a_{ij} = |\{e \mid e \in E, f_G(e) = (v_i, v_j)\}|$

***Дефиниция*:** Последователността от върхове $v_{i_0}, v_{i_1}, \ldots, v_{i_l}$ на крайния ориентиран мултиграф наричаме *маршрут*,ако $\forall$ j = 0, 1, 2,…, l-1, $\exists\, e \in E$ такова, че $f_G(e) = (v_{i_j}, v_{i_{j+1}})$. Числото l наричаме дължина на този маршрут. В случая, когато $v_{i_0} = v_{i_l}$ маршрута наричаме *контур*.

***Теорема*:** Нека G(V, E, f_G) е краен ориентиран мултиграф и нека *M* = $||a_{ij}||$ е матрицата му на съседства. Нека $M^k = ||a_{ij}^{(k)}||$ е *к*-та степен на *M* при целочислено умножение на матрици. Тогава $a_{ij}^{(k)}$ е броят на маршрутите с дължина *к* от v_i до v_j в крайния ориентиран мултиграф G.

***Дефиниция*:** Последователността от върхове $v_{i_0}, v_{i_1}, \ldots, v_{i_l}$ на крайният неориентиран граф G(V, E) наричаме *път* в G от v_{i_0} до v_{i_l}, ако $(v_{i_j}, v_{i_{j+1}} \in E)$, $\forall$ j и $v_{i_{j-1}} \neq v_{i_{j+1}}$. Броят l на ребрата наричаме *дължина* на пътя. Ще считаме че съществува тривиален път с дължина 0 от всеки връх v_i до v_i (така отчитаме наличието на примки в неориентирания граф, но им даваме тежест 0 при образуването на дължината на пътя).

За всеки краен неориентиран граф G(V, E) дефинираме релацията $P_G \subseteq V \times V$, така $(v_i, v_j) \in P_G$ тогава и само тогава когато съществува път в G от v_i до v_j.

***Теорема*:** Релацията P_G е релация на еквивалентност.

Дефиниция: Върховете от всеки клас на еквивалентност $V' \subseteq V$ на релацията P_G индуцират в G(V, E) по един подграф, който наричаме *свързана*

компонента на G. Графът G(V, E) се нарича *свързан*, ако $\forall\, v_i, v_j \in V$, $\exists$ път в G от v_i до v_j, т.е. G има точно една свързана компонента.

Забележка: Дефинията е приложима и за ориентирания случай, както и за мултиграф, със замяна на понятието път с маршрут.

Дървета

Дефиниция: Свързан граф D(V, E) без цикли наричаме *дърво*. Несвързан граф без цикли – гора.

Дефиниция:

а) Графът D({r}, {}) наричаме *дърво с корен* r (*кореново дърво*). Единственият връх r е и единствен *лист* на това дърво.

б) Нека D(V, E) е дърво с корен r $\in V$ и листа $l_1, l_2, ..., l_r$. Нека $u \in V$ и $w \notin V$. Тогава $D'(V', E') = D'(V \cup \{w\}, E \cup \{(u, w)\}$ е също дърво с корен r. Ако $u = l_i$, $1 \le i \le$ r, тогава листа на D' са $l_1, l_2, ..., l_{i-1}, w, l_{i+1}, ..., l_r$. В противен случай листа на D' са $l_1, l_2, ..., l_r, w$

в) Няма други коренови дървета

Забележка: Операцията която приложихме в индуктивната стъпка на горната дефиниция, наричаме *присъединяване на връх*. Кореновите дървета са неориентирани графи, но от дефиницията получаваме **неявна ориентация** на всяко ребро – от върха, към който присъединяваме (баща) към присъединения връх (син). Затова, при записване на ребрата на кореново дърво ще поставяме на първо място в наредената двойка бащата, а на второ – сина.

Теорема: Всяко дърво с корен е дърво

Доказателство: 1. Всяко кореново дърво е свързан граф. а) D({r}, {}) е очевидно свързан. б)Нека D(V, E) е свързан. в) $D'(V', E') = D'(V \cup \{w\}, E \cup \{(u, w)\}$, също е свързан, защото между всеки два върха от *V* има път (от ИП), а от *w* до всеки друг връх има път с първо ребро (*w,u*).

2 Кореновото дърво няма цикли. а) D({r}, {}) очевидно няма цикли. б) Нека D(V, E) няма цикли. в) Допускаме, че D(V', E') има цикли. Тогава (u,w) участва в цикъл. Но d(w) = 1 $\Rightarrow$ противоречие. Сл. и D(V', E') няма цикли.

Теорема: Нека D(V, E) е дърво с корен r, тогава $|V| = |E| + 1$

Доказателство: а) За D({r}, {}) – очевидно, понеже $|V| = 1$, $|E| = 0$. б) Допускаме че за кореновото дърво D(V, E) е в сила $|V| = |E| + 1$. в) За

кореновото дърво $D'(V',E')$, $V'=V\cup\{w\}$, $E'=E\cup\{(r,w)\}$ $\Rightarrow |V'|=|V|+1$, $|E'|=|E|+1 \Rightarrow |V'|-|E'|=|V|-|E|=1 \Rightarrow |V'|=|E'|+1$

Теорема: За всяко дърво D(V, E), съществува единствен път между всеки два върха на D.

Теорема: Нека D(V, E) е кореново дърво и $(v_i,v_j)\notin E$, тогава в $G(V,E\cup\{(v_i,v_j)\})$ има единствен цикъл.

Дефиниция: Нека G(V, E) е граф, а $D\ (V,E')$, $E'\subseteq E$ е дърво. Тогава D наричаме *покриващо дърво* на G.

Теорема: Графът G(V, E) има покриващо дърво $\Leftrightarrow$ когато е свързан.

Обхождане на графи

Различните задачи върху графи могат да изискват специфичен алгоритъм за обхождане на графа. Съществуват обаче техники, които водят до построяване на сходни алгоритми за задачи, които на практика съществено се различават. Такива техники ще наричаме *алгоритмични схеми*. Под *обхождане* на граф ще разбираме процедура, която систематически, по определени правила „посещава" (разглежда) всички върхове на графа.

Обхождане в ширина: Съществено за тази алгоритмична схема е понятието *ниво на обхождане*, което представлява подмножеството на множеството от върхове на свързания граф G(V, E). В резултат на обхождането в ширина, V се разбива на нива $L_0, L_1, ..., L_k$ по следният начин)

1) Избираме начален връх i_0 на обхождането. $L_0=\{i_0\}$ и обявяваме върха i_0 за „обходен", и $i=0$.

2) Ако $L_0\cup L_1\cup ...\cup L_i=V$ прекратяваме обхождането, иначе преминаваме към стъпка 3).

3) Нека сме обходили върховете от нивата $L_0, L_1, ..., L_i$.

   $L_{i+1}=\{v \mid v$ е „необходен" и $\exists w\in L_i, (v,w)\in E\}$.

   Обявяваме всички върхове $v\in L_{i+1}$ за „обходени", $i=i+1$ и преминаваме към стъпка 2).

Забележка: Под „обхождане" на връх, разбираме някакви специфични действия върху разглеждания връх, зависещи от конкретната задача.

За илюстрация на алгоритмичната схема „обхождане в ширина” ще построим следният алгоритъм:

Алгоритъм за построяване на покриващо дърво (в ширина):

Нека е даден свързан граф G(V, E). Целта е да построим покриващо дърво D(V, E') на G.

Алгоритъм:

1) Коренът на покриващото дърво r ще изберем за начален връх на обхождането. Затова L_0={r}. Образуваме $D_0(V_0, E_0), V_0 = L_0, E_0 = \emptyset$, и $i = 0$.

2) Ако $V_i = V$ - край и резултатът е D_i, иначе продължваме със стъпка 3).

3) Нека $D_i(V_i, E_i)$ е дървото построена след i-тата стъпка и L_i са върховете от i-то ниво. $L_{i+1} = \{v \mid v \notin V_i, \exists w \in L_i, (w, v) \in E\}$. Образуваме дървото $D_{i+1}(V_{i+1}, E_{i+1}), V_{i+1} = V_i \cup L_{i+1}, E_{i+1} = E_i \cup \{(w, v) \mid w \in E_i, v \in E_{i+1}\}$, $i = i + 1$ и преминаваме към стъпка 2).

Обхождане в дълбочина: За тази схема основни понятия са *текущ връх* t и *баща на върха* t – p(t).

1) Избираме i_0 за начален връх, $t = i_0$, а p(t) е неопределен.

2) Търсим „необходим” връх, който е съседен на t.

а) Ако има такъв връх v, тогава („стъпка напред”): p(v)=t, $t = v$ и обявяваме v за обходен. Преминаваме към стъпка 2).

б) Няма такъв връх:

- Ако $t \neq i_0$ (началният връх) („стъпка назад”) t = p(t) и преминаваме към стъпка 2).

- Ако $t = i_0$ (началният връх) – край на обхождането.

Схемата за обхождане в дълбочина ни предписва да „опитваме” стъпки напред (в дълбочина) докато това е възможно, и при невъзможност да направим стъпка назад и отново да опитваме стъпка напред. Отново ще решим същата задача (построяване на покриващо дърво с корен r на свързан граф), но с алгоритмичната схема обхождане в дълбочина. Под „обхождане” на връх е естествено да разбираме включване на този връх в дървото.

Алгоритъм за построяване на покриващо дърво (в дълбочина):

Нека ни е даден свързан граф G(V, E). Целта е да построим покриващо дърво D(V, E') на G.

Алгоритъм:

1) Коренът на покриващото дърво r ще изберем за начален връх на обхождането. Образуваме $D_0(V_0, E_0), V_0 = \{r\}$, $E_0 = \emptyset$, $t = r$, $i = 0$ и p(t) е неопределен

2) Нека е построено дървото $D_i(V_i, E_i)$ Търсим $v \notin V_i$, такъв че $(t, v) \in E$:

а) Ако има такъв v, строим $D_{i+1}(V_{i+1}, E_{i+1}), V_{i+1} = V_i \cup \{v\}, E_{i+1} = E_i \cup \{(t, v)\}$, p($v$) = t, t = v, $i = i+1$ и преминаваме към стъпка 2).

б) Иначе (няма такъв връх).

б.1) Ако $t \neq r$, t = p(t) и преминаваме към стъпка 2).

б.2) Иначе – край. Търсеното дърво е $D_i(V_i, E_i)$

Ойлерови обхождания: Път в свързан мултиграф G(V, E), който минава еднократно през всяко ребро на мултиграфа, наричаме *Ойлеров път*. Ако Ойлеровият път има начало и край, които съвпадат то тогава той се нарича *Ойлеров цикъл*. Мултиграф, ребрата, на който образуват Ойлеров цикъл наричаме Ойлеров граф.

Теорема (Л. Ойлер): Свързаният мултиграф G(V, E) е Ойлеров $\Leftrightarrow$ всеки връх на G е с четна степен.

Доказателство:

$\Rightarrow$) Нека ребрата на G(V, E) образуват Ойлеров цикъл, тогава всеки връх има четна степен, понеже при обхождане на мултиграфа по Ойлеровия цикъл, на всяко ребро, „влизащо" във v_i съответства „излизащо" такова (различно от първото, според дефиницията за път), а цикълът съдържа всички ребра точно по един път.

$\Leftarrow$) Нека мултиграфът G(V, E) е свързан и такъв, че всеки връх има четна степен. Ще опишем процедура която построява Ойлеров цикъл като използва всички ребра от графа точно по един път.

1) Нека r е произволен връх в мултиграфа. Обявяваме го за текущ t.

2) Докато е възможно, строим път по следното правило, от върха t, в който се намираме, преминаваме към някой съседен връх v по „необходено" ребро. Използваното ребро обявяваме за „обходено" а върхът v за текущ(t = v).

Нека да допуснем че когато правилото от тази стъпка не е приложимо то $t \neq r$. Тогава ще се окаже че степента на t е нечетна (ние сме във връх от който не можем да продължим), тъй като когато сме влезли за последно във t сме обявили реброто за „обходено, а при предишни пъти когато сме минавали през този връх сме отбелязвали като обходени по 2 негови ребра – противоречие с четността на степените на върховете. Следователно когато 2) приключи то $t = r$ и всички на този етап обходени ребра образуват цикъл C.

3) Ако цикълът C съдържа всички ребра – край на обхождането. Построен е Ойлеровият цикъл C.

4) Вече имаме цикъл с начало и край r, но имаме оставащи ребра, които образуват цикли. Трябва да включим тези цикли в нашият цикъл C)

Ако премахнем всички „посетени" ребра от мултиграфа G(V, E), то отново в графа G(V, E) всички върхове са с четна степен. Търсим такъв връх r' в намерения вече сикъл C, от който излиза поне едно необходено ребро. Ако допуснем че няма такъв връх, и не всички ребра са обходени, ще получим противоречие със свързаността на мултиграфа. Преминаваме към стъпка 2) с начален връх $r = r'$ и построяваме нов цикъл C'. Разширяваме C с C' във върха r' и преминаваме към стъпка 3).

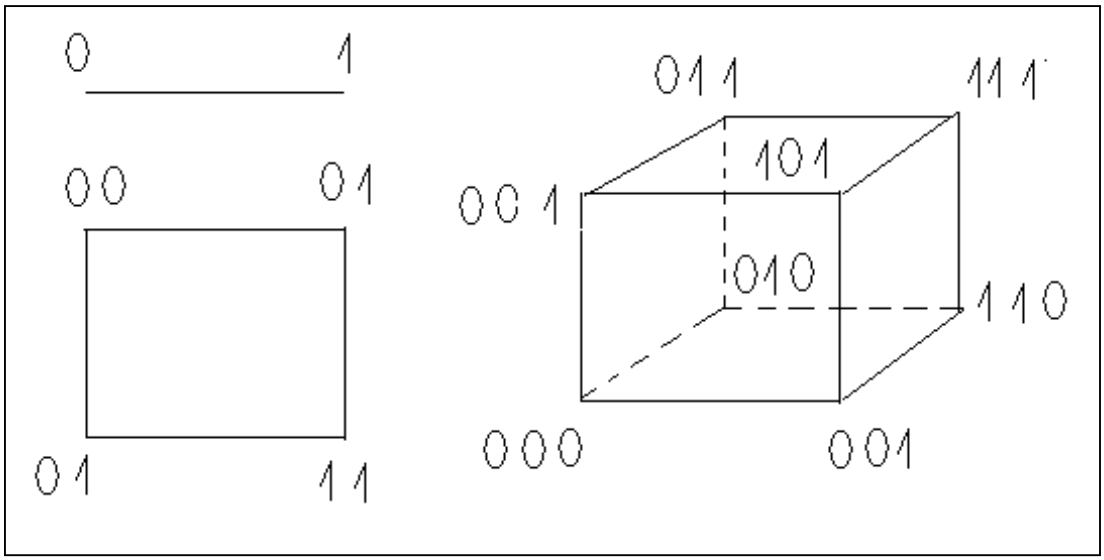
Следствие: Свързаният мултиграф G(V, E) има Ойлеров път $\Leftrightarrow$ когато точно два върха са с нечетна степен.

Хамилтонови обхождания: Път в свързания граф G(V, E), минаващ еднократно през всеки връх на графа, наричаме *Хамилтонов път*. Ако началото и краят на пътя съвпадат, той се нарича *Хамилтонов цикъл*. Граф, който съдържа Хамилтонов цикъл се нарича *Хамилтонов граф*.

На пръв поглед дефиницията на Ойлеров и Хамилтонов граф е подобна, но всъщност това не е така. Не е известно необходимо и достатъчно условие за един граф да е Хамилтонов. Също така не е известен бърз алгоритъм, който да проверява дали зададен граф е Хамилтонов.

Ще разгледам интересен клас графи, за които е лесно да се докаже, че са Хамилтонови.

Дефиниция: Графът $B_2^n(\{0,1\}^n, E_n)$ за $n \geq 1$ и $E_n = \{(\alpha, \beta) \mid \alpha$ и β се различават само в една позиция (1 бит)$\}$, наричаме *n-мерен булев* куб.



Теорема: Графът B_2^n е Хамилтонов за $n \geq 2$

Доказателство: Индукция по n

a) При $n = 2$ е очевидно. Пътят 00,01,11,10,00 е Хамилтонов цикъл.

b) Допускаме че твърдението е вярно за n и нека $\alpha_0, \alpha_1, .., \alpha_{2^n-1}, \alpha_0$ е Хамилтонов цикъл в B_2^n.

c) Построяваме следната последователност от върхове $0\alpha_0, 0\alpha_1, .., 0\alpha_{2^n-1}, 1\alpha_{2^n-1}, 1\alpha_{2^n-2}, ..., 1\alpha_0, 0\alpha_0$, която съдържа всички върхове на (n+1)-мерния куб. Очевидно $0\alpha_i$ и $0\alpha_{i+1}$ при $i = 0,1,...,2^n - 2$ се различават само в една позиция (от индукционното предположение) и образуват ребра в B_2^n, аналогично за $1\alpha_i$ и $1\alpha_{i-1}$ при $i = 2^n - 1, 2^n - 2, ..., 1$. Така също $(0\alpha_{2^n-1}, 1\alpha_{2^n-1})$ е ребро, както и $(1\alpha_0, 0\alpha_0)$. Следователно посочената последователност е цикъл, който минава точно по веднъж през всеки връх.