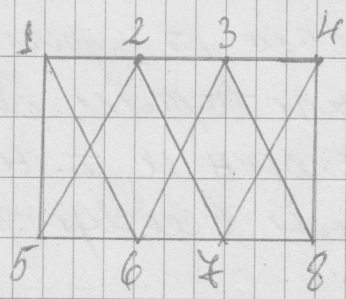
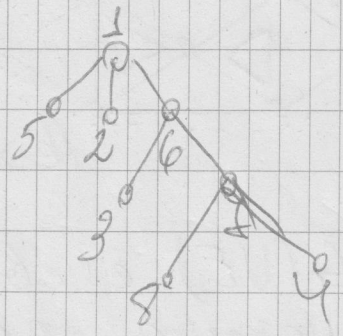
ДМ Лист 6 07.05

Алгоритъм за намиране на покриващо дърво в свързване

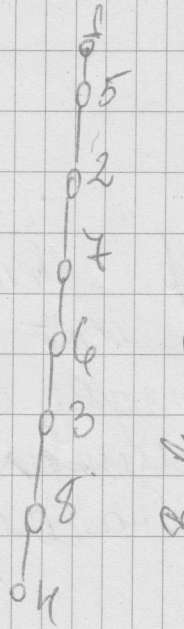
Нека $G(V,E)$ е свързан граф
$p(v)$ - предшественик на v, t - текущ връх

1) Нека r е произволен връх (той може да е корен)
$p(r)$ - неопределен и $t := r$, $c = 0$, $D_c = (\{r\}, \emptyset)$
2) Проверяваме дали съществува v - съсед на t ($(t,v) \in E$ означава t съсед на v)
а) Ако такова v съществува, тогава такова v и го фиксираме $p(v) = t$, $t := v$
$V_{c+1} = V_c \cup \{t\}$; $E_{c+1} = E_c \cup \{p(t), t\}$
$D_{c+1} = (V_{c+1}, E_{c+1})$, $c = c+1$. Премини към 2
б) Ако такова v не съществува то проверяваме дали $t \neq r$
б1) Ако $t \neq r$ то $t := p(t)$. Премини към 2
б2) Ако $t = r$ то край. $D_c(V_c, E_c)$ е покриващо дърво в графа





В ширина

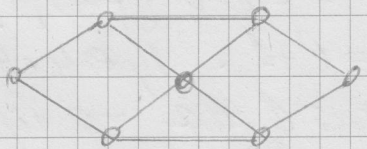


В дълбочина

Ойлерови и Хамилтонови графи

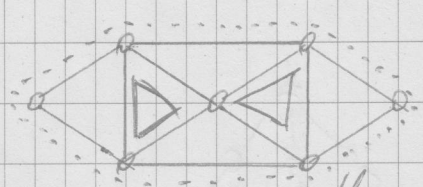
Деф Нека $G(VE)$ е свързан граф. Ойлеров път се нарича път в който всяко ребро на графа участва само веднъж. Ако допълнително началото и края съвпадат то този път се нарича Ойлеров цикъл

Деф $G(VE)$ е Ойлеров граф $\Leftrightarrow$ G притежава Ойлеров цикъл



Тв. Един свързан граф $G(VE)$ е Ойлеров $\Leftrightarrow$ всички върхове на графа имат четна степен (Н върх или степен брой ребра излизащи от него)

Док



Тръгваме от произволен връх и търсим съседен необходим и т. н. докато не обходим всички ребра

Деф Нека $G(VE)$ е свързан граф. Хамилтонов път се нарича път в който минава през всеки връх точно веднъж. Хамилтонов цикъл се нарича хамилтонов път в който началото и края съвпадат

Деф Хамилтонов граф се нарича такъв свързан граф който има хамилтонов цикъл

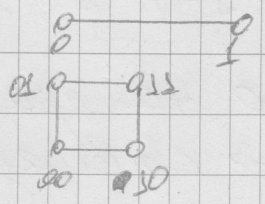
$V_n = \{\alpha \mid \alpha \in \mathbb{Z}_2^n\}$ - наредени двойки n-торки

$E_n = \{\{\alpha, \beta\} \mid \delta(\alpha\beta) = 1\}$ при $\delta(\alpha\beta) = \sum_{i=1}^{n} |a_i - b_i|$

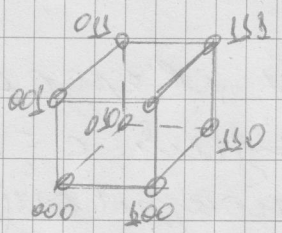
Граф $B_n(V_n, E_n)$ - n-мерен куб

едномерен

двумерен



тримерен



Тв За всяко $n \geq 2$ B_n е хамилтонов граф

Оптимизационни задачи за графи

Нека $G(VE)$ е граф и $c: E \to \mathbb{R}_+$ $([0, +\infty])$ казваме c е цена (ценова функция) и/или тегло (тегловa функция)

Деф Нека $D(VE')$ е покриващо дърво за графа G - Тегло на покриващото дърво D се нарича сумата $\sum_{e \in E'} c(e) = c(D)$

казваме че покриващото дърво $D_0(V, E_0)$ е минимално (максимално), ако за всяко покриващо дърво $D'(V, E')$ е изпълнено $c(D_0) \leq c(D')$ ($\max c(D_0) \geq c(D')$)

Деф Нека $v_{i_0} - v_{i_e}$ е път в $G(V, E)$. Цена на пътя означаваме с c се $c(v_{i_0} - v_{i_e}) =$

$$\sum_{j=1}^{e} c(v_{i_{j-1}}, v_{i_j})$$

Ако за всяко ребро на графа $c(e) = 1$, тогава цената на пътя съвпада с дължината на пътя

Задача За даден свързан граф G и ценова функция $c: E \to \mathbb{R}_+$ да се намери минимално покриващо дърво

Задача За даден свързан граф G и фиксиран връх v да се намери минималният път от v до всеки връх на графа

ТВ За всеки свързан граф $G(V, E)$ покриващото дърво с минимална цена дава минималните пътища в графа ($c(e) = 1$ за всяко ребро в графа)

Алгоритъм на Прим за минимално покриващо дърво

Нека $G(VE)$ е свързан граф и $c: E \to \mathbb{R}_+$ е ценова функция

1) Нека z е произволен връх
$V_0 = \{z\}$, $E_0 = \emptyset$, $D_0(V_0, E_0)$, $e = 0$
2) Търсим ребро (v, w) т.ч. $v \in V_e$ а $w \notin V_e$ и $c(v,w)$ е най-малко, които изпълняват тези свойства
а) Ако има то $V_{e+1} = V_e \cup \{w\}$, $E_{e+1} = E_e \cup \{(v,w)\}$ $D_{e+1}(V_{e+1}, E_{e+1})$, $e = e+1$. Преминаваме към 2
б) Ако няма то $D_e(V_e, E_e)$ е търсеното покриващо дърво - Край

Алгоритъм на Крускал

Нека $G(VE)$ е свързан граф и $c: E \to \mathbb{R}_+$
Подреждаме ребрата по тегло $e_0 \dots e_m$, като $c(e_0) \leq c(e_1) \leq \dots \leq c(e_m)$

1) $D_1(\{v_1\}, \emptyset) \dots D_n(\{v_n\}, \emptyset)$ са текущата редица от дървета
2) Проверяваме дали редицата се състои от едно дърво
а) Ако да то това дърво е търсеното покриващо дърво
б) Ако не то търсим първото e_i, което не принадлежи на никое дърво и $e_i = (v_{i1}, v_{i2})$ $v_{i1} \in V'$, $v_{i2} \in V''$ за $D'(V', E')$ и $D''(V'', E'')$ двете дървета в редицата. Получаваме $D'''(V' \cup V'', E' \cup E'' \cup \{v_{i1} v_{i2}\})$ Преминаваме към 2

Нека $G(V,E)$ е свързан граф и $c: E \to \mathbb{R}_+$

Разширението $c^*: V^2 \to \mathbb{R}_+ \cup \{\infty\}$ се дава по следния начин

$$c^*(v_i, v_j) = \begin{cases} c(v_i, v_j) & \text{ако } (v_i, v_j) \in E \\ \infty, & \text{ако } (v_i, v_j) \notin E \end{cases}$$

$V = \{v_0 \dots v_n\}$, $s = v_0$

$dist[j]$ - най-късото текущо разстояние от $v_0 \to v_j$

$prev[j]$ - текущ предшественик на връх v_j

Алгоритъм на Дейкстра

1) $dist[j] = c^*(0, j)$ за $j > 0$

$dist[0] = 0$, $U = \{v_0\}$ където

U - множеството от временно намерените най-късо разстояние; $prev[0]$ - неопределен

2) Повтаряме $n-1$ пъти следната процедура

Търсим най-малкото $dist[i]$, $U = U \cup \{v_j\}$

За всяко $v_k \notin U$ $dist[k] = \min\{c^*(0,k), dist[i] + c^*(j,k)\}$; $prev[k] = j$