ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅՈՒՆ

ԿՐԹՈՒԹՅԱՆ, ԳԻՏՈՒԹՅԱՆ, ՄՇԱԿՈՒՅԹԻ ԵՎ ՍՊՈՐՏԻ

ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ ՀԱՅԱՍՏԱՆԻ ԱԶԳԱՅԻՆ ՊՈԼԻՏԵԽՆԻԿԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆ

ՏԵՂԵԿԱՏՎԱԿԱՆ ԵՎ ՀԱՂՈՐԴԱԿՑԱԿԱՆ ՏԵԽՆՈԼՈԳԻԱՆԵՐԻ և ԷԼԵԿՏՐՈՆԻԿԱՅԻ ԻՆՍՏԻՏՈՒՏ ԻՆՏԵԳՐԱԼ ՍԽԵՄԱՆԵՐԻ ԵՎ ՀԱՄԱԿԱՐԳԵՐԻ ՄՇԱԿՄԱՆ ԾՐԱԳՐԱՅԻՆ ՄԻՋՈՑՆԵՐԻ ԱՄԲԻՈՆ

ԿՈՒՐՍԱՅԻՆ ԱՇԽԱՏԱՆՔ

Առաջադրանքը տրվեց ՝ 09.02.2023
Կուրսային աշխատանքի պաշտպանությունը ՝ 02.06.2023

Խումբ՝ SS019-U

Առարկա՝ Դիսկրետ մաթեմատիկա

Թեմա՝ Միակցման կետերի որոնման ալգորիթմի մշակում և ծրագրային իրացում

Դասախոս՝ Սարգսյան Գարեգին

Ուսանող՝ Բաբայան Ալվարդ

Բովանդակություն

Ներածություն

Դիցուք $V = \{ v_1, v_2, \ldots, v_p\}$ ցանկացած ոչ դատարկ վերջավոր բազմություն է, և դիցուք $V^{(2)}$-ը V բազմության տարրերի բոլոր ոչ կարգավոր զույգերի բազմությունն է: Ենթադրենք, որ $E \subseteq V^{(2)}$:

(V, E) կարգավոր զույգին կանվանենք գրաֆ, և այն կնշանակենք G-ով:

$G = (V,E)$ գրաֆի V բազմության տարրերին կանվանենք գրաֆի *գագաթներ*, իսկ E բազմության տարրերին՝ *կողեր*:

Դիցուք $G = (V, E)$ գրաֆ է, $u,v \in V$ և $e, e' \in E$:

u և v գագաթներին կանվանենք հարևան,եթե $u,v \in E$:

u գագաթին և e կողին կանվանենք կից, եթե $u \in e$:

e և e′ տարբեր կողերը կանվանենք հարևան,եթե գոյություն ունի $v \in V$ այնպես, որ v կից է e –ին և e′-ին:

Եթե $G = (V,E)$ գրաֆում $V = \{v_1, \ldots , v_n\}$ և $E = \{e_1, \ldots , e_m\}$, ապա այդ գրաֆին համապատասխանեցնենք $n \times n$ կարգի $A(G) = (\alpha_{ij})_{n\times n}$ մատրիցը հետևյալ կերպ

$$\alpha_{ij} = \begin{cases} 1, & \text{եթե } v_i \text{ և } v_j \text{ հարևան են,} \\ 0, & \text{հակառակ դեպքում:} \end{cases}$$

$A(G)$ մատրիցը կանվանենք G գրաֆի *հարևանության մատրից:* Նկատենք, որ ցանկացած i- ի համար ($1 \le i \le n$) $a_{ii} = 0$, և ցանկացած i, j-ի համար ($1 \le i, j \le n$) $a_{ij} = a_{ji}$: v գագաթին կից կողերի բազմությունը՝$j_G(\{v\})$- ն, կնշանակենք $j_G(v)$-ով:

G գրաֆը կոչվում է լրիվ, եթե նրանում ցանկացած երկու գագաթ հարևան են:

H գրաֆը կոչվում է G գրաֆի ենթագրաֆ և կգրենք $H \subseteq G$, եթե $V(H) \subseteq V(G)$ և $E(H) \subseteq E(G)$: Հակառակ դեպքում, կգրենք $H \not\subseteq G$:

Դիցուք $G = (V, E)$ գրաֆ է: $c(G)$-ով նշանակենք G գրաֆի կապակցված բաղադրիչների քանակը:

G գրաֆի (u_0,u_k)- շրջանցումը կանվանենք u_0-ից u_k ճանապարհի կամ (u_0,u_k)- ճանապարհ, եթե u_0, u_0,-ը,..., u_0 u_0-ն G գրաֆի զույգ առ զույգ տարբեր կողեր են: Եթե P-ն G գրաֆի ճանապարհ է, ապա |P|-ով կնշանակենք այդ ճանապարհի երկարությունը , այսինքն՝ այդ ճանապարհի մեջ առկա կողերի քանակը:

G գրաֆը կանվանենք կապակցված, եթե նրա ցանկացած երկու u և v գագաթների համար G գրաֆում գոյություն ունի (u,v) - ճանապարհ:

G գրաֆի v գագաթը կոչվում է միակցման կետ, եթե c(G-v)>c(G):

Խնդրի դրվածքը

Տրված G(V, E) վերջավոր գրաֆի միակցման կետերի որոնման ալգորիթմի մշակում և ծրագրային իրացում:

Խնդրի լուծում

Դիցուք G = (V, E) գրաֆ է.
Գագաթը կոչվում է գրաֆի հոդակապման կետ, եթե գագաթի և հարակից եզրերի հեռացումը անջատում է գրաֆը:

G գրաֆում միակցման կետերը գտնելու համար կատարենք հետևյալ քայլերը`

- Ներմուծել ֆայլի միջոցով գրաֆի գագաթների քանակը և գրաֆի գագաթների միջև կապերը:
- Գտնում է գրաֆի միակցման կետերը օգտագործելով ծառի ալգորիթմը:
    - Ստուգելով գրաֆի ծառը, նրա ճյուղերը, ծնողները և երեխաները:
    - Գտնում է գրաֆի միակցման կետերը օգտագործելով ծառը:
    - Նկարում է գրաֆը գունավորելով միակցման կետերը և գրաֆի մյուս գագաթներ համապատասխանաբար կանաչ և կարմիր:

Ալգորիթմի հիմնական իմաստը հետևյալն է `

Մեկ առ մեկ հեռացնել բոլոր գագաթները և տեսնել, թե արդյոք գագաթի հեռացումը առաջացնում է անջատված գրաֆիկ:

Օգտագործել եմ DFS (խորությամբ փնտրում)

DFS-ում հետևում ենք գագաթներին ծառի տեսքով, որը կոչվում է DFS ծառ: DFS ծառի մեջ u գագաթը մեկ այլ v գագաթի մայրն է, եթե v-ն հայտնաբերվել է u-ի կողմից:

DFS ծառի մեջ u գագաթը հոդակապման կետ է, եթե հետևյալ երկու պայմաններից մեկը ճիշտ է:

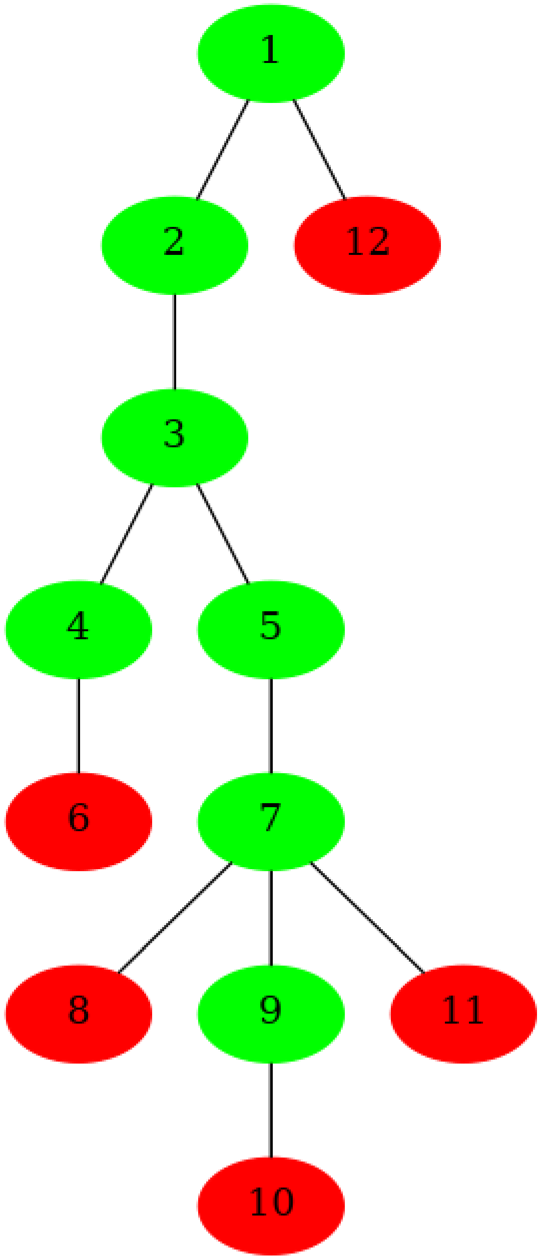
- u-ը DFS ծառի արմատն է, և այն ունի առնվազն երկու երեխա:
- u-ը DFS ծառի արմատը չէ և այն ունի v երեխա այնպես, որ v-ով արմատավորված ենթածառի ոչ մի գագաթ չունի u-ի DFS ծառի նախնիներից մեկի հետևի եզրը:

DFS ծառի տերևը երբեք չի կարող լինել միակցման կետ:

Ստուգելու համար, արդյոք u-ն DFS ծառի արմատն է և ունի առնվազն երկու երեխա: Յուրաքանչյուր գագաթի համար հաշվենք երեխաներին: Եթե ներկայումս այցելած u գագաթը արմատ է (parent[u]-ը NULL է) և ունի ավելի քան երկու երեխա, տպենք այն: Երկրորդ դեպքը կարգավորելու համար, երբ u-ը DFS ծառի արմատը չէ, և այն ունի v երեխա, այնպես, որ v-ով արմատավորված ենթածառի ոչ մի գագաթ չունի ետևի եզր դեպի DFS ծառի նախնիներից որևէ մեկը, պահպանեք զանգվածում գագաթների հայտնաբերման ժամանակը պահելու համար: Յուրաքանչյուր u հանգույցի համար պարզենք ամենավաղ այցելած գագաթը (գագաթը նվազագույն հայտնաբերման ժամանակով), որին կարելի է հասնել u-ով արմատավորված ենթածառից: Այսպիսով, մենք պահպանում ենք լրացուցիչ զանգված low[] այնպես, որ. low[u] = min(disc[u], disc[w]) , Այստեղ w-ը u-ի նախահայրն է և կա հետևի եզր` u-ի որոշ ժառանգներից մինչև w:

Խնդրի լուծման ծրագրային նկարագրությունը C++ ծրագրավորման լեզվով կարող եք տեսնել անցնելով հղումը` **https://github.com/babayanal/discrete_math :**

Խնդրի լուծում



Եզրակացություն

Օգտագործելով ծառի գաղափարը և DFS ալգորիթմը գտել եմ գրաֆի միակցման կետերը:
Ժամանակային բարդությունը DFS-ի համար O(V+E) է , որտեղ V-ն գրաֆի գագաթների թիվն է, E-ն՝ կողերի։

Օգտագործված գրականության ցանկ

Պ.Ա. Պետրոսյան, Վ.Վ. Մկրտչյան, Ռ.Ռ. Քամալյան - Գրաֆների տեսություն

Ռ.Ն. Տոնոյան - Դիսկրետ մաթեմատիկայի դասընթաց