ԲՈՎԱՆԴԱԿՈՒԹՅՈՒՆ

ՆԵՐԱԾՈՒԹՅՈՒՆ

Ժամանակակից քաղաքային միջավայրը ներկայացնում է բարդ և բազմաշերտ համակարգ, որի արդյունավետ գործունեությունը մեծապես կախված է ճանապարհային ցանցի ճիշտ պլանավորումից։ Քանի որ մեծ քաղաքներում երթևեկության ծավալը անընդհատ աճում է՝ բնակչության թվի, ավտոմոբիլների քանակի, ինչպես նաև ծառայությունների մատուցման ոլորտի ընդլայնման հետևանքով, մեծ կարևորություն են ստանում այն մեթոդները, որոնք հնարավորություն են տալիս օպտիմալ ձևով կազմակերպել մարդկանց և տրանսպորտային միջոցների տեղաշարժը։ Քաղաքային տրանսպորտի արդյունավետ պլանավորումը չի սահմանափակվում միայն ճանապարհների կառուցմամբ կամ երթուղային գծերի թվով․ այն պահանջում է հստակ հաշվարկված, մաթեմատիկական հիմքով օպտիմալացման մեխանիզմներ։

Տրանսպորտային համակարգը կարելի է դիտարկել որպես օբյեկտների և նրանց միջև կապերի ցանց։ Այս մոտեցումը առավել ամբողջական է դառնում գրաֆերի տեսության կիրառմամբ, որի միջոցով կանգառները ներկայացվում են որպես գագաթներ, իսկ ճանապարհային հատվածները՝ որպես կողեր։ Այդպիսի մոդելավորումն ապահովում է ոչ միայն համակարգի կառուցվածքային պարզեցում, այլ նաև հնարավորություն է տալիս կիրառել հաշվարկային արդյունավետ մեթոդներ՝ եզրակացություններ կատարելու, օպտիմալ ուղիներ գտնելու և պլանավորման որոշումներ ընդունելու համար։ Քաշված գրաֆերի միջոցով հնարավոր է վերլուծել ճանապարհների երկարությունները, տեղափոխման ժամանակները, երթևեկության ինտենսիվությունը և այլ չափանիշներ, որոնք չափազանց կարևոր են տրանսպորտային խնդիրների լուծման համար։

Տրանսպորտային համակարգերի պլանավորման մեջ առանձնահատուկ նշանակություն ունի ամենակարճ ուղու խնդիրը։ Այն հնարավորություն է տալիս գնահատել, թե ինչ կերպ կարելի է քաղաքային տարածքում ապահովել առավել արագ և արդյունավետ տեղաշարժ՝ նվազագույն ծախսով։ Այս խնդիրը լայնորեն ուսումնասիրված է մաթեմատիկայում և հաշվարկային գիտություններում, և դրա համար մշակվել են բազմաթիվ ալգորիթմներ՝ Դեյքստրայի, Բելլման-Ֆորդի, Ֆլոյդ-Ուորշալ, A* և այլ մեթոդներ։Թեև դրանցից բոլորը ունեն կարևոր տեսական արժեք, քաղաքային տրանսպորտի պլանավորման խնդիրներում առավել արդյունավետ է Dijkstra-ի ալգորիթմը, որն աշխատում է դրական քաշերով գրաֆերի վրա և ապահովում է արագ և ճշգրիտ արդյունքներ։

Այս աշխատանքի նպատակը գրաֆերի տեսության և ամենակարճ ուղու որոշման ալգորիթմների հիման վրա երթուղիների պլանավորման մեխանիզմի մշակումն է և դրա կիրառումը քաղաքային տրանսպորտային ցանցի օրինակով: Աշխատանքի ընթացքում ուսումնասիրվել են մաթեմատիկական մոդելավորման սկզբունքները, դիտարկվել են գրաֆի կառուցվածքային բաղադրիչները, ներկայացվել են ամենակարճ ուղու հաշվարկման համար նախատեսված հիմնական ալգորիթմները և դիտարկվել է դրանց կիրառելիությունը իրական խնդիրների լուծման համար: Բացի տեսական մասից, աշխատանքում ներառված է նաև ծրագրային իրականացում Pyhton լեզվով, որն ապահովում է մոդելի գործնական փորձարկումը: Մոդելը կարող է ծառայել ինչպես ուսումնական, այնպես էլ կիրառական նպատակներով՝ հանդիսանալով ուղեցույց ճանապարհային ծրագրավորման և երթուղային հաշվարկների համար:

Այսպիսով, աշխատանքը միավորում է թե՛ տեսական, թե՛ գործնական բաղադրիչներ՝ նպատակ ունենալով ցույց տալ, որ գրաֆերի տեսությունը և դրանց վրա հիմնված հաշվարկային մեթոդները կարող են հանդիսանալ շատ արդյունավետ գործիք քաղաքային տրանսպորտի պլանավորման մեջ:

ԳԼՈՒԽ 1. ԳՐԱՖՆԵՐԻ ՏԵՍԱԿԱՆ ՀԻՄՔԵՐԸ

1.1. Գրաֆների տեսության հիմնական գաղափարները

Գրաֆների տեսությունը մաթեմատիկայի և հաշվարկային գիտությունների այն բաժիններից է, որը տրամադրում է հիմնարար միջոցներ բարդ համակարգերի կառուցվածքն ու կապերը նկարագրելու համար։ Գրաֆի գաղափարը սկիզբ է առել դեռևս XVIII դարում Լեոնարդ Օյլերի աշխատություններից, սակայն վերջին տասնամյակներում այն լայնորեն կիրառվում է ինֆորմացիոն համակարգերում, տրանսպորտային մոդելավորումում, սոցիալական ցանցերի վերլուծության մեջ, էլեկտրական սխեմաների նախագծման ժամանակ և այլ բազմաթիվ ոլորտներում։

Գրաֆը ներկայացվում է որպես տարրերի զույգ՝

$$G = (V, E),$$

որտեղ V բազմությունը գագաթների, իսկ E բազմությունը՝ կողմերի բազմությունն է։ Յուրաքանչյուր գագաթ ունի իր նշանակությունը, իսկ յուրաքանչյուր կող՝ իր ֆունկցիան։ Գրաֆների տեսության գլխավոր առավելություններից մեկը կայանում է նրանում, որ այն տալիս է հստակ մաթեմատիկական մոտեցում իրական օբյեկտների միջև եղած հարաբերությունները ներկայացնելու համար։

Քաղաքային տրանսպորտի մոդելավորման տեսանկյունից գագաթներն են կանգառները՝ «Կենտրոն», «Արաբկիր», «Զեյթուն», «Նոր Նորք», «Ավան», «Մալաթիա» և այլն, իսկ կողերը դրանց միջև եղած ճանապարհային հատվածներն են, որոնք ունեն որոշակի երկարություն կամ ճանապարհի անցման միջին ժամանակ։

Այսպիսով, գրաֆը հանդիսանում է ոչ միայն տեսական գործիք, այլ նաև կիրառական մոտեցում՝ նպատակ ունենալով ներկայացնել, ուսումնասիրել և օպտիմալացնել երթուղիները իրական ցանցերում։

1.2. Գրաֆի կառուցվածքը. գագաթներ և կողմեր

Գագաթը գրաֆի հիմնական միավորն է՝ այն օբյեկտը, որը մասնակցում է կապերի համակարգում։ Գագաթները կարող են ներկայացնել տարբեր իմաստային միավորներ՝ կանգառներ, խաչմերուկներ, շենքեր, սարքեր, օգտատերեր և այլն։ Քանակով գագաթները նշվում են | V |։

Կողը գագաթների միջև կապ է, որը կարող է լինել ուղղորդված կամ ոչ ուղղորդված։

Ոչ ուղղորդված կողերում կապը երկկողմանի է։

Ուղղորդված կողերում կապը միակողմանի է, և գագաթների միջև անվանական տարբերություն կա։

Քաղաքային տրանսպորտում կողմերը հաճախ դիտարկվում են որպես երկկողմանի, քանի դեռ տվյալ ճանապարհահատվածը թույլ է տալիս երկու ուղղությամբ երթևեկություն։

Կողերին տրվում է **քաշ**՝ թվային արժեք, որը նկարագրում է գագաթների միջև եղած կապի «արժեքը»՝ հեռավորություն, ժամանակ, ծախս կամ այլ չափանիշ։ Այս արժեքների հիման վրա լուծվում են ամենակարճ ուղու խնդիրները։

1.3. Կշռված գրաֆներ և նրանց նշանակությունը

Կշռված գրաֆը այն մոդելն է, որտեղ յուրաքանչյուր կող ունի որոշակի քաշ

$$w(u, v) \in \mathbb{R}.$$

Սա հատկապես կարևոր է երթուղիների պլանավորման համար, քանի որ իրական ճանապարհային հատվածները տարբեր երկարություններ ունեն։

Օրինակ՝

- «Կենտրոն»–«Արաբկիր»՝ 3.2 կմ
- «Արաբկիր»–«Զեյթուն»՝ 4.1 կմ
- «Զեյթուն»–«Նոր Նորք»՝ 2.8 կմ

Կշռված գրաֆը թույլ է տալիս լուծել հետևյալ հարցերը․

- ո՞րն է ամենակարճ ուղին A կետից B կետ,
- ո՞ր ընտրությունն է առավել արդյունավետ,
- ինչպե՞ս հաշվարկել ընդհանուր ճանապարհի երկարությունը,
- արդյո՞ք գոյություն ունի ավելի նպաստավոր տարբերակ։

Այսպիսով, կշռված գրաֆները հանդիսանում են երթուղիների պլանավորման հիմնական մաթեմատիկական հիմք։

1.4. Ուղու սահմանում և ընդհանուր քաշ

Ուղին գագաթների հաջորդականություն է՝

$$P = (v_1, v_2, \ldots, v_k),$$

որտեղ յուրաքանչյուր v_i–v_{i+1} զույգ միացված է կողով։

Ուղու ընդհանուր քաշը որոշվում է որպես նրա բոլոր կողմերի քաշերի գումարը․

$$L(P) = \sum_{i=1}^{k-1} w(v_i, v_{i+1}).$$

Օրինակ՝ «Կենտրոն» → «Արաբկիր» → «Զեյթուն» ուղու երկարությունը կլինի

$$3.2 + 4.1 = 7.3 \text{ կմ.}$$

Եթե A–B ուղիներ կան մի քանի տարբերակներով, ապա ամենակարճ խնդիրը ընտրում է նվազագույն արժեք ունեցողը։

1.5. Կապակցվածություն և բաղադրիչներ

Գրաֆը կոչվում է կապակցված, եթե յուրաքանչյուր երկու գագաթների միջև գոյություն ունի ուղի։ Կապակցվածությունը շատ կարևոր պայման է երթուղիների պլանավորման մեջ, քանի որ եթե գոյություն ունեն անջատված հատվածներ, ապա հնարավոր է որոշ կանգառներ լցնել չմատչելի։

1.6. Գրաֆի ներկայացման եղանակներ

Գրաֆը համակարգչում ներկայացնելու համար օգտագործվում են երկու հիմնական եղանակներ․

1) Հարևանության մատրիցա

Քառակուսի մատրիցա $A = [a_{ij}]$, որտեղ

$a_{ij} = w(i, j)$, եթե կա կող,

$a_{ij} = \infty$(կամ 0), եթե չկա։

Առավելություն՝ արագ հասանելիություն։
Թերություն՝ շատ հիշողություն է պահանջում՝ $O(V^2)$։

2) Հարևանության ցուցակ

Յուրաքանչյուր գագաթի համար պահպանվում է նրա հարևանների ցուցակը։

Առավելություններ՝

- արդյունավետ մեծ, բայց սակավաճյուղ գրաֆների դեպքում,
- ավելի քիչ հիշողություն։

Թերություններ՝

- որոշ ալգորիթմների համար բարդ է արագ ներդաշնակեցնել։

Քաղաքային ցանցերի մեծ մասի համար երկրորդ տարբերակը ավելի նպատակահարմար է։

1.7. Գրաֆային մոդելի կիրառումը տրանսպորտում

Տրանսպորտային ցանցերը բնականորեն համապատասխանում են գրաֆային ներկայացմանը։
Գագաթները կանգառներն են, իսկ կողմերը՝ ճանապարհները։

Այս գրաֆը կարող է ունենալ հետևյալ խտացում․

քաշեր՝ ճանապարհի երկարություններ,

ուղղորդում՝ միակողմանի փողոցների դեպքում,

լրացուցիչ պարամետրեր՝ լուսացույցերի ժամանակ, խցանումներ, տրանսպորտի հաճախություն։

Լավ մոդելավորված գրաֆը թույլ է տալիս հաշվարկել․

- օպտիմալ երթուղի,
- հաջորդական կանգառներ,
- ընդհանուր ճանապարհի երկարություն,
- կամ այլ չափանիշներով պլանավորումներ։

1.8. Ամենակարճ ուղու խնդրի տեղը գրաֆների տեսությունում

Կշռված գրաֆներում ամենակարևոր խնդիրներից մեկը ամենակարճ ուղու խնդիրն է։
Այն ունի մեծ կիրառական նշանակություն՝

- տրանսպորտի օպտիմալացում,
- GPS և նավիգացիա,
- ինտերնետային երթուղուցում,
- էլեկտրական ցանցեր.

Այս խնդրի լուծման համար մշակվել են մի շարք ալգորիթմներ, որոնցից ամենատարածվածներն են.

Դեյքստրայի,

Բելլաման-Ֆորդի,

Ֆլոյդ-Ուորշալ,

A* (խիստ կիրառական աշխարհագրական խնդիրներում):

Սրանք ուսումնասիրելուց հետո հնարավոր է ստեղծել մոդել, որն աշխատում է ցանկացած տրանսպորտային ցանցում:

1.9 Հիմնական սահմանումներ

- Դիցուք $V = \{v_1, v_2, \ldots, v_n\}$ վերջավոր բազմություն է և $E \subseteq \{(v_i, v_j)/v_i \neq v_j, v_i, v_j \in V\}$, որտեղ (v_I, v_j) ոչ կարգավորված զույգ է:
- $G = (V, E)$ գրաֆը կոչվում է լրիվ, եթե նրա կամայական երկու գագաթները հարևան են (կող են կազմում):
- Գրաֆը կոչվում է հարթ, եթե այն կարելի է այնպես պատկերել հարթության վրա, որ ցանկացած կող չունենա ինքնահատում և ցանկացած երկու կողեր չունենան ընդհանուր կետեր, բացի գագաթներից:
- $\bar{G} = (V, E')$ գրաֆը կոչվում է $G = (V, E)$ գրաֆի լրացում, եթե ամեն մի v և v'

  գագաթների համար $(v \neq v')$ տեղի ունի $(v, v') \in E' \leftrightarrow (v, v') \notin E$:
- $G_1 = (V_1, E_1)$ և $G_2 = (V_2, E_2)$ գրաֆները անվանում են իզոմորֆ և նշանակում են $G_1 \cong G_2$, եթե գոյություն ունի այնպիսի 1-1 արտապատկերում՝ $f: V_1 \rightarrow V_2$ բազմությունների միջև, որ ցանկացած իրարից տարբեր $v_i, v_j \in V_1$ գագաթների համար տեղի ունի հետևյալ պայմանը.
$$(v_i, v_j) \in E_1 \leftrightarrow (f(v_i), f(v_j)) \in E_2:$$
- $G = (V, E)$ գրաֆում գագաթների և կողերի $v_1, (v_1, v_2), v_2, (v_2, v_3), \ldots, v_{k-1}, (v_{k-1}, v_k), v_k$ (1) հաջորդականությունը, որտեղ $(v_i, v_{i+1}) \in E$ և $i = 1,2, \ldots, k-1$,

կոչվում է v_1 և v_k գագաթները միացնող երթուղի:

- Եթե (1) հաջորդականության մեջ $v_i = v_k$, ապա երթուղին կոչվում է փակ:
- Կրկնվող կողեր չպարունակող փակ երթուղին կոչվում է ցիկլ:
- Գրաֆը կանվանենք համիլտոնյան գրաֆ, եթե այն պարունակում է համիլտոնյան ցիկլ:
- Կասենք, որ գրաֆում գոյություն ունի համիլտոնյան ճանապարհ, եթե -ում գոյություն ունի պարզ ճանապարհ, որն անցնում է -ի բոլոր գագաթներով:
- Եթե այդ պարզ ճանապարհը պարզ ցիկլ է, ապա այն կանվանենք համիլտոնյան ցիկլ:
- G գրաֆը կանվանենք *էյլերյան* գրաֆ, եթե այն պարունակում է էյլերյան ցիկլ:
- Կասենք, որ **G** գրաֆում գոյություն ունի *էյլերյան ճանապարհ*, եթե G-ում գոյություն ունի ճանապարհ, որը պարունակում է **G-**ի բոլոր գագաթները և կողերը:
- Եթե այդ ճանապարհը ցիկլ է, ապա այն կանվանենք *էյլերյան ցիկլ*:
- G գրաֆը էյլերյան է այն և միայն այն դեպքում, երբ G-ն կապակցված գրաֆ է և այդ գրաֆի կողերի բազմությունը կարելի է տրոհել կողերով չհատվող պարզ ցիկլերի:

ԳԼՈՒԽ 2. Ամենակարճ ուղու հաշվարկի ալգորիթմների տեսական հիմքերը

2.1. Ամենակարճ ուղու խնդիրների ընդհանուր նկարագրությունը

Ամենակարճ ուղու խնդիրները հանդիսանում են գրաֆների տեսության և հաշվարկային մաթեմատիկայի ամենակարևոր և լայն կիրառվող խնդիրներից։ Քանի որ բազմաթիվ իրական համակարգեր կարող են ներկայացվել որպես գագաթների և նրանց միջև կապերի մոդել, ապա օպտիմալ ուղու որոնումը դառնում է կենտրոնական հետազոտական ուղղություն՝ սկսած տրանսպորտային ուղիների հաշվարկումից մինչև ինտերնետային երթուղում, նավիգացիա, մատակարարման շղթաների կառավարում, շարժական ռոբոտների կառավարում և տնտեսական գործընթացների օպտիմալացում։

Ամենակարճ ուղու խնդիրն ունի մի քանի հիմնական ձևակերպումներ․

1. Մեկ աղբյուրից մինչև մնացած բոլոր գագաթներ․
2. Մեկ զույգ գագաթների միջև ամենակարճ ուղի․
3. Բոլոր գագաթների զույգերի միջև ամենակարճ ուղիներ

Տրանսպորտային ցանցերի համար հիմնական հետաքրքրությունը ներկայացնում է հենց առաջին ձևը՝ մեկ կանգառից հասնել մնացածներին հնարավորինս նվազագույն ժամանակով կամ հեռավորությամբ։ Այս դեպքում կիրառելի են այն ալգորիթմները, որոնք աշխատում են դրական քաշերով գրաֆներում։

Քանի որ քաղաքային տրանսպորտի դեպքում բացասական երկարություններ գոյություն չունեն, ապա խնդիրն էապես պարզվում է և ուղղվում դեպի ամենաարդյունավետ մեթոդները՝ հատկապես **Դեյքստրայի** ալգորիթմը և նրա տարբերակները։

2.2. Խնդրի մաթեմատիկական ձևակերպումը

Տրված է կշռված գրաֆ՝

$$G = (V, E),$$

որտեղ յուրաքանչյուր կողի վրա տրված է դրական քաշ՝

$$w(u, v) > 0.$$

Պետք է գտնել ուղի

$$P = (v_1, v_2, \ldots, v_k)$$

որով՝

1. $v_1 = s$— մեկնարկային գագաթ,
2. $v_k = t$— նպատակային գագաթ,
3. Գոյություն ունեն եզրեր $(v_i, v_{i+1}) \in E$,
4. Ուղու ընդհանուր քաշը

$$L(P) = \sum_{i=1}^{k-1} w(v_i, v_{i+1})$$

նվազագույնն է բոլորը հնարավոր ուղիների մեջ:

Այս մաթեմատիկական ձևակերպումը հիմք է հանդիսանում ամենակարճ ուղու ալգորիթմների տեսական և ծրագրային մշակման համար:

2.3. Օպտիմալության սկզբունքները

Ամենակարճ ուղու խնդիրը հիմնված է երկու հիմնական սկզբունքի վրա.

1) Օպտիմալության դինամիկ սկզբունք

Եթե $P = (s \to t)$ամենակարճ ուղին է, ապա դրա յուրաքանչյուր ենթուղի նույնպես ամենակարճ է:

Այսինքն՝ եթե

$$s \to u \to t$$

ամենակարճ ճանապարհն է, ուրեմն նաև

$$s \to u$$

ամենակարճ է:

2) Մոնոտոնիկության սկզբունք

Դրական քաշերի պայմաններում ուղու ընդհանուր արժեքը երբեք չի նվազում, երբ երկարացնում ենք ճանապարհը:

Այս սկզբունքներն ապահովում են, որ լուծումը կարելի է կառուցել նախորդ հաշվարկների վրա հիմնվելով:

2.4. Դեյքստրայի ալգորիթմը

Դեյքստրայի ալգորիթմը համարվում է ամենաարդյունավետ և ամենատարածված մեթոդը դրական քաշերով գրաֆներում ամենակարճ ուղու որոշման խնդրի լուծման համար: Այն հիմնված է քաշերի աստիճանական նվազեցման գաղափարի վրա, որտեղ ամեն քայլում ընտրվում է այն գագաթը, որը տվյալ պահին լավագույն (նվազագույն) գնահատումն ունի:

2.4.1.Դեյքստրայի ալգորիթմի կիրառությունը

1. Ճանապարհային և նավիգացիոն համակարգեր

- GPS, Maps, Waze և այլ նավիգացիոն ծրագրերը օգտագործում են Դեյքստրայի ալգորիթմը՝ արագ հաշվարկելու քաղաքի փողոցների և մայրուղիների միջև ամենակարճ երթուղիները:
- Ճանապարհների քաշերը համարվում են՝ հեռավորություններ, ժամանակ, վառելիքի ծախս կամ տրանսպորտային ծանրաբեռնվածություն:

2. Ցանցային երթուղավորող պրոտոկոլներ

Դեյքստրան հիմք է մի շարք ցանցային տեխնոլոգիաների համար.

- **OSPF (Open Shortest Path First)**՝ ինտերնետային երթուղավորման ամենավստահելի պրոտոկոլներից:
- Ալգորիթմը օգնում է սարքերին ընտրել տվյալները փոխանցելու ամենաարագ ուղին՝ նվազեցնելով ցանցի բեռնվածությունը:

3. Տրանսպորտի և լոգիստիկայի օպտիմալացում

- Առաքում կատարող կազմակերպությունները (UPS, Glovo, Bolt) օգտագործում են այս ալգորիթմը՝ պլանավորելու դրոնների, մեքենաների կամ առաքիչների շարժը:
- Օգնում է նվազեցնել՝ հեռավորություն, ժամանակ, ծախսեր:

4. Էներգետիկ համակարգեր

- Օգտագործվում է էլեկտրական ցանցերում՝
  առավել արդյունավետ էներգիայի հոսքերի հաշվարկման համար:

2.4.2. Ալգորիթմի քայլերը

1. **Սկզբնական կարգավորում.**

Ալգորիթմի մեկնարկին յուրաքանչյուր գագաթի սահմանվում է նախնական «հեռավորություն»՝

- Սկզբնական գագաթի հեռավորությունը սահմանվում է 0,
- Մնացած բոլոր գագաթների հեռավորությունները՝ անվերջություն (∞):

Միաժամանակ ստեղծվում է երկու բազմություն.

- Անավարտ (չայցելված) գագաթներ,
- Ավարտված (այցելված) գագաթներ, որոնք սկզբում դատարկ են:

2. **Ամենափոքր հեռավորություն ունեցող գագաթի ընտրություն.**

Անավարտ բազմությունից ընտրվում է այն գագաթը, որի ներկայիս հեռավորության արժեքը ամենափոքրն է: Այդ գագաթը հանդիսանում է տվյալ պահին «ամենամոտ» գագաթը սկզբնակետից: Մինչև այդ բազմությունը դատարկ է

3. **Հարակից գագաթների հեռավորությունների թարմացում**

Ընտրված գագաթի բոլոր հարևանների համար հաշվվում է նոր հնարավոր հեռավորությունԵթե նոր հաշվված արժեքը փոքր է, քան արդեն գրանցվածը, ապա այն փոխարինվում է: Այս քայլը ապահովում է, որ ալգորիթմը պահպանում է միայն ամենակարճ հայտնի ուղիները:

4. **Գագաթի ավարտում**

Երբ գագաթի բոլոր հարևանները ստուգվում են, այն նշվում է որպես ավարտված, և այլևս չի վերանայվում:
Ալգորիթմը երաշխավորում է, որ այդ գագաթի համար գտնված արժեքը վերջնական և օպտիմալ է:

5. **Ալգորիթմի շարունակություն**

Քայլերը (2–4) կրկնվում են այնքան ժամանակ, մինչև բոլոր գագաթները ավարտված լինեն, կամ չմնա գագաթ, որը հասանելի է սկզբնական գագաթից:

2.4.3. Հաշվարկային բարդությունը

Բարդությունը կախված է օգտագործվող տվյալների կառուցվածքից․

Իրականացում	**Բարդություն**
Առանց heap	$O(V^2)$
Min-heap կամ priority queue	$O(E\log V)$

2.5. Բելլաման-Ֆորդի ալգորիթմը

Բելլաման-Ֆորդի ալգորիթմը նախատեսված է այն դեպքերի համար, երբ գրաֆում կարող են լինել բացասական քաշեր: Թեպետ տրանսպորտային ցանցերում բացասական արժեքներ չեն հանդիպում, այնուամենայնիվ այս ալգորիթմը տեսական կարևորություն ունի:

2.5.1. Բելլաման-Ֆորդի ալգորիթմի կիրառությունը

1. Բացասական քաշերով գրաֆներում ամենակարճ ուղիների որոնում

Եթե գրաֆում կան եզրեր, որոնց քաշը բացասական է (օր.՝ շահույթի, զեղչի, ռիսկի նվազեցման մոդելներ),

- ֆինանսական հաշվարկներ, որտեղ շահույթը ներկայացվում է որպես **բացասական ծախս**
- ռիսկային ուղիների մոդելավորում

2. Բացասական ցիկլերի հայտնաբերում

- ֆինանսական շուկաներում
- օպտիմալացման խնդիրներում
- մեքենայական ուսուցման որոշ մոդելներում

3. Ծախսերի նվազեցման խնդիրներ

- արտադրական գործընթացների օպտիմիզացում

- էներգետիկ համակարգեր

4. Ռոբոտաշինություն

- Երբ ռոբոտի ճանապարհին կան «վնասի» կամ «ռիսկի» գործոններ, որոնք ներկայացվում են բացասական արժեքներով:

2.5.2. Ալգորիթմի քայլերը

1. Մուտքային տվյալներ

Տրված է ուղղորդված կամ չուղղորդված գրաֆ՝

$$G = (V, E)$$

որտեղ՝

- V– գագաթների բազմություն
- E– եզրերի բազմություն
- Յուրաքանչյուր եզր ունի քաշ՝

$$w(u, v) \in \mathbb{R}$$

Գոյություն ունի սկզբնական գագաթ՝

$$s \in V$$

2. Սկզբնականացում

Բոլոր գագաթների համար նախ սահմանում ենք.

$$\mathrm{d(v)} = \infty, \forall \mathrm{v} \in \mathrm{V}$$

Սկզբնական գագաթի համար.

$$\mathrm{d(s)} = 0$$

Այստեղ d(v)— գագաթ v-ի նկատմամբ հայտնաբերված ամենակարճ հեռավորությունն է:

3. Թուլացման քայլ

Յուրաքանչյուր եզրի համար $(u, v) \in E$
ստուգում ենք՝ արդյո՞ք կարելի է ավելի կարճ ճանապարհ ստանալ.

եթե $d(u) + w(u, v) < d(v)$, ապա թարմացնում ենք:
$d(v) := d(u) + w(u, v)$

Այս գործողությունը կոչվում է **թուլացում**:

4. Հիմնական ցիկլը — կատարվում է |V|−1 անգամ

Ալգորիթմը կրկնում է թուլացման քայլը

$$|V| - 1$$

անգամ, քանի որ ցանկացած ամենակարճ ուղի առավելագույնը ունի $|V| - 1$ եզր։

5. Բացասական ցիկլի հայտնաբերումը

Վերջում կատարում ենք ևս մեկ լրացուցիչ անցում.
եթե որևէ եզր դեռ կարող է «թուլացվել», ապա գրաֆում կա **բացասական քաշով ցիկլ**, և ամենակարճ ուղին **չի կարող միանշանակ սահմանվել**:

Ստուգումը՝

$$\exists (u, v) \in E: d(u) + w(u, v) < d(v) \Rightarrow \text{բացասական ցիկլ կա}$$

Ալգորիթմի ամփոփ տեսքը

1. Սահմանել՝

$$d(s) = 0, d(v) = +\infty \text{ մնացած } v\text{–երի համար}$$

2. $|V| - 1$անգամ՝ բոլոր (u, v)-երի համար կատարել.

$$d(v) = \min(d(v),\ d(u) + w(u, v))$$

3. Ստուգել բացասական ցիկլի գոյությունը:

2.5.3. Բարդությունը

$$O(VE)$$

Սա զգալիորեն դանդաղ է Դեյքստրայի համեմատ:

2.6. Ֆլոյդ-Ուորշալի ալգորիթմը

Այս ալգորիթմը լուծում է բոլոր գագաթների զույգերի միջև ամենակարճ ուղիների հաշվարկը:
Այն հիմնված է դինամիկ ծրագրավորման վրա և ունի հետևյալ գաղափար.
յուրաքանչյուր փուլում փորձել բարելավել ուղիները միջանկյալ գագաթի միջոցով:

Շնորհիվ իր պարզության՝ Ֆլոյդ-Ուորշալի ալգորիթմը հարմար է միջին չափի գրաֆների համար և տալիս է ամբողջական Ամենակարճ ուղու խնդրի որոշման մատրիցա:

2.6.1. Ֆլոյդ–Ուորշալի ալգորիթմի կիրառություններ

1. Ճանապարհային և տրանսպորտային համակարգեր

- քարտեզներում և նավիգացիոն համակարգերում
- երթուղիների պլանավորման և օպտիմալացման մեջ
- քաղաքային տրանսպորտի սիմուլյացիաներում
- բեռնափոխադրումների օպտիմալ ցանցերի ստեղծման ժամանակ

2. Ցանցային երթուղում

- տվյալների փոխանցման օպտիմալ ուղիներ որոշելու համար
- բազմաթիվ հանգույցներից տվյալների փոխանցման ժամանակ հաշվարկելու լավագույն ուղիները

3. Կիբերանվտանգություն

- վտանգավոր կամ վնասաբեր գործողությունների ցիկլերը հայտնաբերելու համար
- բացասական ազդեցությամբ ցիկլերի վերլուծության մեջ

4. Օպտիմիզացիայի խնդիրներ

- արտադրական գործընթացների մոդելավորում
- ռիսկերի նվազեցման և շահույթի բաշխման հաշվարկներում

2.6.2. Ալգորիթմի քայլերը

1. Մուտքային տվյալները

Տրված է կշռված ուղղորդված գրաֆ՝

$$G = (V, E)$$

որտեղ՝

- $V = \{1,2,\ldots,n\}$— գագաթների բազմություն
- E— եզրերի բազմություն
- Յուրաքանչյուր եզր ունի քաշ՝

$$w(i,j) \in \mathbb{R}$$

2. Սկզբնական մատրիցա

Ստեղծում ենք հեռավորությունների մատրիցա՝

$$D^{(0)} = (d_{ij}^{(0)})$$

Պայմանները՝

- Եթե $i = j$, ապա

$$d_{ij}^{(0)} = 0$$

- Եթե կա եզր $i \to j$, ապա

$$d_{ij}^{(0)} = w(i,j)$$

- Հակառակ դեպքում

$$d_{ij}^{(0)} = +\infty$$

3. Հիմնական գաղափարը

Ալգորիթմը ստուգում է՝
ակարո՞ղ է միջանկյալ գագաթ k-ը կատարել քայլ, որը դարձնում է
i-ից դեպի jքայլը ավելի կարճ:

Մաթեմատիկորեն՝

$$d_{ij}^{(k)} = \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)})$$

Այսինքն՝
կամ արդեն հայտնի ուղին $i \to j$ է ամենակարճը,
կամ $i \to k \to j$–ը ավելի կարճ է:

4. Բացասական ցիկլերի հայտնաբերում

Եթե վերջում ունենանք

$$d_{ii} < 0$$

որևէ i-ի համար, ապա գրաֆում կա **բացասական քաշով ցիկլ**:

2.6.2. Բարդությունը

$$O(V^3)$$

2.7. A* ալգորիթմը

A* ալգորիթմը օգտագործում է էվարիստիկ ֆունկցիա «ճանապարհ + արժեք»՝ կանխատեսելու նպատակային գագաթին մոտեցման արժեքը:

Հիմնական գաղափարը․

$$f(v) = g(v) + h(v)$$

որտեղ՝

- $g(v)$— ստույգ արժեք է (անցած ճանապարհի քաշ),
- $h(v)$— էվարիստիկ (մոտավոր գնահատում մինչև վերջին կետ):

Եթե էվարիստիկը ընդունելի է (չի գերագնահատում), ապա ալգորիթմը երաշխավորում է օպտիմալ լուծում:

2.7.1. A* ալգորիթմի կիրառությունը

1. Նավիգացիոն համակարգերում (GPS, Maps, Waze)

- A* ալգորիթմը լայնորեն կիրառվում է երթուղիների հաշվարկման համար:
- Այն օգտագործում է heuristic՝ օրինակ ուղիղ գծով հեռավորություն, որպեսզի ավելի արագ գտնի նպատակակետը՝
ի տարբերություն Դեյքստրայի, որը ստուգում է բոլոր հնարավոր տարբերակները:

2. Ռոբոտատեխնիկա և ինքնավար համակարգեր

- Ինքնավար ռոբոտները և մեքենաները օգտագործում են A*՝ օպտիմալ ուղին գտնելու և բախումները կանխելու համար։

3. Ավիացիա և օդային երթուղիներ

- Օգնում է հաշվարկել օդային կորիդորներով ամենաարդյունավետ շարժը։
- Օգտագործվում է նաև drone–ների երթուղու պլանավորման մեջ։

4. Քարտեզների և տարածական վերլուծության ծրագրեր

Գործածվում է GIS համակարգերում՝

- տարածքային օպտիմալ ուղիներ գտնելու,
- տարածքների միջակայքային հաշվարկների համար։

2.7.2. Ալգորիթմի քայլերը

1) **Սկիզբի պատրաստություն**

- Գրել սկզբնական կետը (S) և վերջնակետը (G)։
- Յուրաքանչյուր կետ ունի երկու արժեք՝
  - $g(n)$– ճանապարհի իրական արժեքը սկզբից մինչ այդ կետը
  - $h(n)$– հիւմանական գնահատականը այդ կետից վերջնակետը
  - $f(n) = g(n) + h(n)$

2) **Բաց և փակ ցանկեր**

- **Բաց ցանկ** – կետերը, որոնք պետք է ուսումնասիրվեն։ Սկզբում պարունակում է միայն S կետը։
- **Փակ ցանկ** – կետերը, որոնք արդեն ուսումնասիրվել են։ Սկզբում դատարկ է։

3) **Կետ ընտրել**

- Բացել ցանկից ընտրել այն կետը, որի $f(n)$ամենափոքրն է։

4) **Ստուգել վերջնակետը**

- Եթե ընտրված կետը G-ն է, ալգորիթմը ավարտվում է, և ճանապարհը վերականգնվում է՝ հետևելով ծնողական նշաններին, որոնք ցույց են տալիս, որ կետից է ճանապարհը եկել տվյալ կետ :

5) **Տարբերակները լայնացնել**

- Ընտրված կետի հարևան կետերը հաշվարկել:
- Ամեն հարևան կետի համար հաշվարկել $g(n)$, $h(n)$և $f(n)$արժեքները:

6) **Թարմացնել ցանկերը**

- Եթե հարևանը չկա բաց կամ փակ ցանկում → ավելացնել բաց ցանկում և նշել ծնողը:
- Եթե հարևանը բաց ցանկում արդեն կա, բայց նոր ճանապարհը ավելի լավ է (նվազագույն $g(n)$) → թարմացնել ծնողը և արժեքները:
- Եթե հարևանը փակ ցանկում է, նոր ճանապարհը լավ է → տեղափոխել բաց ցանկ և թարմացնել:

7) **Կրկնել**

- Վերադառնալ 3-րդ քայլին, մինչև բաց ցանկը դատարկ չլինի կամ վերջնակետը գտնվի:
-

2.7.3. Բարդությունը

1. Ժամանակային բարդություն
   - Լավագույն դեպքում (առաջարկվող heuristics-ը շատ լավ է)՝ O(d), որտեղ d-ն նպատակակետին հասնելու ուղի երկարությունն է
   - Միջին դեպքում՝ O(b^d), որտեղ b-ն միջին ճյուղավորվողությունն է, d-ն ուղի երկարությունը
   - Վատագույն դեպքում (heuristics-ն օգնում չի տալիս)՝ O(|E| + |V| log |V|)
2. Տարածքային բարդություն
   - O(|V|)՝ բաց և փակ ցանկերը պահելու համար

2.8. Ալգորիթմների համեմատությունը տրանսպորտային ցանցերում

Տրանսպորտային ցանցերում ալգորիթմների ընտրությունը կախված է ճանապարհային տվյալների բնույթից։

Համեմատական աղյուսակ.

Ալգորիթմ	Դրական քաշեր	Բացասական քաշեր	Բարդություն	Տիպական օգտագործում
Դեյկստրա	✓	✗	$O(E \ \backslash log\ V)$	✓
Բելման–Ֆորդ	✓	✓	$O(V\ E)$	✗
Ֆլոյդ–Ուորշալ	✓	✓	$O(V^3)$	Մեծ չափի ամբողջական մատրիցային հաշվումներ
A*	✓	✗	Կախված է էվրիստիկից	✓ GPS, քարտեզներ