| Sorsz. | Új él | Halmazok | Bevál. élek sz. |
|---|---|---|---|
| 1 | $B - C$ | $H_1 = \{B, C\}$ | 1 |
| 2 | $B - E$ | $H_1 = \{B, C, E\}$ | 2 |
| 3 | $F - G$ | $H_1,\ H_2 = \{F, G\}$ | 3 |
| 4 | $E - J$ | $H_1 = \{B, C, E, J\},\ H_2$ | 4 |
| 5 | $I - L$ | $H_1,\ H_2,\ H_3 = \{I, L\}$ | 5 |
| 6 | $K - L$ | $H_1,\ H_2,\ H_3 = \{I, K, L\}$ | 6 |
| 7 | $J - K$ | $H_1 := H_1 \cup H_3 = \{B, C, E, I, J, K, L\},\ H_2$ | 7 |
| 8 | $A - B$ | $H_1 = \{A, B, C, E, I, J, K, L\},\ H_2$ | 8 |
| 9 | $C - D$ | $H_1 = \{A, B, C, D, E, I, J, K, L\},\ H_2$ | 9 |
| 10 | $A - H$ | $H_1 = \{A, B, C, D, E, H, I, J, K, L\},\ H_2$ | 10 |
| 11 | $D - N$ | $H_1 = \{A, B, C, D, E, H, I, J, K, L, N\},\ H_2$ | 11 |
| 12 | $A - E$ | $A \in H_1,\ E \in H_1$ | 11 |
| 13 | $K - N$ | $K \in H_1,\ N \in H_1$ | 11 |
| 14 | $M - N$ | $H_1 = \{A, B, C, D, E, H, I, J, K, L, M, N\},\ H_2$ | 12 |
| 15 | $D - J$ | $D \in H_1,\ J \in H_1$ | 12 |
| 16 | $H - J$ | $H \in H_1,\ J \in H_1$ | 12 |
| 17 | $H - K$ | $H \in H_1,\ K \in H_1$ | 12 |
| 18 | $A - F$ | $H_1 := H_1 \cup H_2$ | 13 |

2.2. táblázat.

## 2.2. A legrövidebb út problémája

DIJSKTRA algoritmus

E fejezetben azzal foglalkozunk, hogy egy hálózat két pontja között hogyan lehet megkeresni a legrövidebb utat. A *legrövidebb út* kifejezést természetesen tágabb értelemben használjuk: jelenthet időben legrövidebb utat ill. legolcsóbb utat is.

Az alapfeladat a következő: adott egy $N$-pontú $(P_1, \ldots, P_N)$ hálózat, melynek éleihez távolságértékek (időtartam, költség, stb.) vannak rendelve. Jelölje $\nu(P_i, P_j)$ a $P_i$ és $P_j$ pontok távolságát $(\nu(P_i, P_j) = \nu(P_j, P_i))$. Keressük a hálózat két adott pontja (kezdő- és célpont) közti legrövidebb utat. Természetesen semmi megszorítást nem jelent, ha úgy sorszámozzuk a pontokat, hogy a kezdőpontot $P_1$ a célpontot pedig $P_N$ jelölje. A feladatot az ún. Dijkstra algoritmus segítségével, két részben oldjuk meg.

### Az algoritmus első része

Jelöljük a $P_1$-ből a $P_i$ pontba vezető legrövidebb út hosszát $t(P_i)$-vel $(i = 1, \ldots, N)$. Az első rész lényege, hogy minden ponthoz meghatározzuk a $t(P_i)$ értéket. Ezt úgy hajtjuk végre, hogy az $n$-edik lépésben $(n = 1, 2, \ldots, N)$ meghatározzuk a kezdőponthoz az n-edik legközelebbi pontot. Ezt a pontot $M_n$-nel jelöljük és $n$-edik megoldott pontnak hívjuk. Az $t(P_i)$ értékeket a rájuk vonatkozó felső becsléseket finomítva kapjuk meg. Kezdeti felső becslésként szimbolikusan $\infty$-t használunk.

Tegyük fel, hogy már meghatároztuk az $M_1, \ldots, M_n$ pontokat, és a hozzájuk vezető legrövidebb utak hosszát és vannak felső becsléseink is néhány nem megoldott pont esetén a hozzájuk vezető legrövidebb út hosszára. Tekintsük most az $M_n$ megoldott pontot és a hozzá tartozó legrövidebb útvonalhosszhoz adjuk hozzá a vele szomszédos még nem megoldott pontok $M_n$-től mért távolságait. Amennyiben az így kapott összeg valamelyik még nem megoldott pontra kisebb lesz, mint az erre a pontra vonatkozó eddigi felső becslés, akkor használjuk ezt az összeget a továbbiakban felső becslésként. Az $M_{n+1}$ az a $P_i$ pont lesz, amelynek a legkisebb a felső becslése és $t(P_i)$ ez a felső becslés lesz. Ha több ilyen pont is van, akkor bármelyiket választhatnánk közülük, de állapodjunk meg abban, hogy mindig a legkisebb sorszámút választjuk ilyenkor. A használt algoritmus tehát a következő. (Az eljárás során $t(P_i)$ a $P_i$-be vezető legrövidebb út hosszának felső becslését jelenti. Ezen értékek egyenként válnak a tényleges $t(P_i)$ értékekké.)

2.3. Algoritmus. (A legrövidebb út meghatározása a Dijkstra algoritmussal.)

*1.* $t(P_i) := \infty$ *(*$i = 1, \ldots, N$*),* $t(P_1) := 0$

*2.* $M_1 := P_1$, $n := 1$.

*3. Válasszunk egy* $M_n$*-nel szomszédos még nem megoldott* $P$ *pontot, amely még nem lett kiválasztva* $M_n$*-hez. Ha nincs ilyen, akkor ugorjunk az 5. pontra.*

*4. Legyen* $\delta := t(M_n) + \nu(M_n, P)$*. Ha* $\delta < t(P)$*, akkor* $t(P) := \delta$*. Vissza a 3. ponthoz.*

*5.* $\gamma := \min_P\{t(P)\}$*, ahol a minimumot az összes még nem megoldott* $P$ *pontra kell venni.*

*6.* $M_{n+1}$ *az a még nem megoldott* $P$ *pont lesz, melyre* $t(P) = \gamma$*. Ha több ilyen* $P$ *is van, akkor vesszük a legkisebb sorszámút.* $n := n + 1$.

*7. Ha* $n \leq N - 1$*, akkor vissza a 3. ponthoz, egyébként ugrás az algoritmus második részére.*

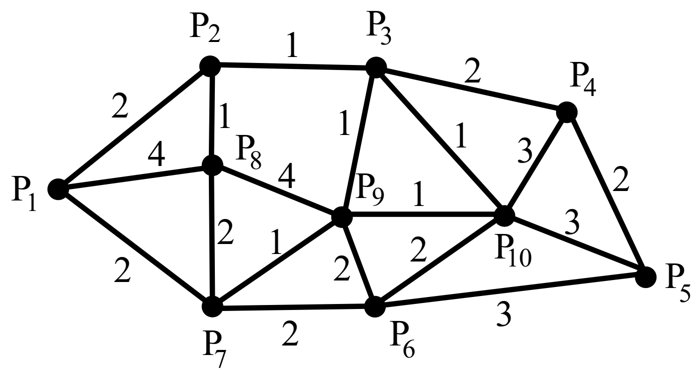
### Az algoritmus második része

A második lépésben az előző pontban kiszámított legrövidebb úthosszak segítségével meghatározzuk a $P_1$-ből $P_N$-be vezető legrövidebb utat. Tegyük fel, hogy a $P_1$-ből $P_N$-be vezető legrövidebb út áthalad a $P_j$ ponton. Ekkor a legrövidebb úton haladva $P_j$-be azon $P_j$-vel szomszédos $P_i$ pontból kell érkeznünk, melyre $\nu(P_i, P_j) = t(P_j) - t(P_i)$. Ha több ilyen $P_i$ pont is van, akkor a legrövidebb út elágazik, azaz a feladatnak több megoldása is lesz. A fent leírt eljárást a $P_N$ ponttól indítva meghatározhatjuk a legrövidebb utat.

A fenti algoritmussal kapcsolatban megjegyezzük, hogy ha $P_N$ helyett másik célpontot választunk, akkor az eljárás első lépése nem fog megváltozni, csak az útvonalmeghatározásnál kell figyelembe venni a megváltozott célpontot. Ezzel szemben, ha a kezdőpontot változtatjuk meg, akkor az összes $t(P_i)$ értéket újra ki kell számolni, mivel az eddigiek a $P_1$ ponttól mért minimális távolságokat adták meg.

## Mintafeladat

Következzék egy feladat, mellyel szemléltetni szeretnénk a Dijkstra algoritmust. Adott 10 város ($P_1, \ldots, P_{10}$) melyek között a 2.5. ábrának megfelelően vannak összekötő utak. Minden úthoz adott az az idő (órában) amely alatt egy szállítmány képes azon az úton végighaladni. Feladatunk az, hogy egy szállítmányt a $P_1$ városból a lehető legrövidebb idő alatt eljuttassunk a $P_{10}$ városba. Melyik útvonalat válasszuk, és minimálisan hány óra alatt lehet a szállítást kivitelezni?




2.5. ábra. *Hálózat a legrövidebb út problémájához*

A megoldás első lépését, azaz a $P_1$ pontból a többi pontba vezető legrövidebb utak hosszát a 2.3. táblázat segítségével számítottuk ki. A táblázat fejlécében a hálózat pontjait tüntettük fel. Első sorában a $P_1$ ponthoz 0 értéket írtunk, mivel a $P_1$-ből a $P_1$-be vezető legrövidebb út hossza nyilvánvalóan nulla ($t(P_1) = 0, M_1 = P_1$). A többi ponthoz vezető utak hosszait egyelőre $\infty$-nel becsültük felülről. A $P_1$ pont tehát megoldott, így tekintenünk kell a $P_1$-gyel szomszédos pontokat ($P_2, P_7$ és $P_8$), és ezek $P_1$-től mért távolságait hozzá kell adnunk $t(P_1)$-hez. Ezek az összegek láthatók a táblázat 2. sorában. Közülük a 2 a legkisebb, de ehhez az értékhez két pont is tartozik, a $P_2$ és a $P_7$. Válasszuk $P_2$-t, mert ennek a kisebb a sorszáma. Ezzel a $P_2$ pont is megoldottá vált ($M_2 = P_2$, $t(P_2) = 2$). A megoldott pontokhoz tartozó legrövidebb utak hosszát vastagon szedtük.

Most már két megoldott pontunk van, $P_1$ és $P_2$. $P_2$ szomszédos a $P_3$ és $P_8$ nem meghatározott pontokkal. Emiatt a táblázat 3. sorába beírjuk a $P_3$ oszlopába a $t(P_2)+\nu(P_2,P_3) = 3$ értéket, mivel az az eddig ott álló $\infty$ becslésnél jobb, valamint a $P_8$ oszlopába a $t(P_2)+\nu(P_2,P_8) = 3$ értéket, mivel az jobb az eddig ott álló 4-es értéknél. A $P_7$ pontra vonatkozó 2-es értéket továbbvisszük a 3. sorba. A 3. sorban a legkisebb elem a kettes, így $M_3 = P_7$ és $t(P_7) = 2$. Az eljárást hasonlóan folytatjuk tovább, míg minden csúcsra meg nem kapjuk a legrövidebb út hosszát. Az egyes csúcsokra ezeket a hosszokat a táblázat megvastagított számai adják. Pl. $t(P_6) = 4$ és $t(P_9) = 3$. Persze számunkra a $t(P_{10}) = 4$ érték fontos, hiszen ez adja meg, hogy a $P_1$ pontból a $P_{10}$ pontba vezető legrövidebb út hossza 4 óra lesz.

Az algoritmus második részében meghatározzuk a legrövidebb útvonalat. $P_{10}$-be összesen öt ponton keresztül érkezhetünk. Mivel $t(P_{10}) - t(P_6) = 4 - 4 = 0 \neq 2 = \nu(P_{10}, P_6)$, ezért a legrövidebb út nem a $P_6$ ponton keresztül éri el $P_{10}$-et. Könnyen ellenőrizhető, hogy a legrövidebb út vagy a $P_9$ ponton keresztül megy ($t(P_{10})-t(P_9) = 4 - 3 = 1 = \nu(P_{10}, P_9)$), vagy a $P_3$ ponton keresztül ($t(P_{10}) - t(P_3) = 4 - 3 = 1 = \nu(P_{10}, P_3)$). Ezt az eljárást tovább folytatva két legrövidebb útvonalat kapunk: $P_1 - P_7 - P_9 - P_{10}$ ill. $P_1 - P_2 - P_3 - P_{10}$.

Összefoglalva tehát, a szállítmány minimális menetideje 4 óra és kétfajta útvonal közül választhatunk: $P_1 - P_7 - P_9 - P_{10}$ ill. $P_1 - P_2 - P_3 - P_{10}$.

|  | $P_1$ | $P_2$ | $P_3$ | $P_4$ | $P_5$ | $P_6$ | $P_7$ | $P_8$ | $P_9$ | $P_{10}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 1. | **0** | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ | $\infty$ |
| 2. |  | **2** | $\infty$ | $\infty$ | $\infty$ | $\infty$ | 2 | 4 | $\infty$ | $\infty$ |
| 3. |  |  | 3 | $\infty$ | $\infty$ | $\infty$ | **2** | 3 | $\infty$ | $\infty$ |
| 4. |  |  | **3** | $\infty$ | $\infty$ | 4 |  | 3 | 3 | $\infty$ |
| 5. |  |  |  | 5 | $\infty$ | 4 |  | **3** | 3 | 4 |
| 6. |  |  |  | 5 | $\infty$ | 4 |  |  | **3** | 4 |
| 7. |  |  |  | 5 | $\infty$ | **4** |  |  |  | 4 |
| 8. |  |  |  | 5 | 7 |  |  |  |  | **4** |
| 9. |  |  |  | **5** | 7 |  |  |  |  |  |
| 10. |  |  |  |  | **7** |  |  |  |  |  |

2.3. táblázat. A mintafeladat megoldása.

## 2.3. Maximális független élrendszer

Először két új fogalom kerül bevezetésre.

2.4. DEFINÍCIÓ. A $\mathcal{G} = (C, E)$ gráfot páros gráfnak nevezzük, ha a csúcspontok $C$ halmaza két diszjunkt nemüres $C_1$ és $C_2$ részhalmazokra bontható úgy, hogy élek csak $C_1$ és $C_2$ között vezetnek.

2.5. DEFINÍCIÓ. Egy gráfban bizonyos élek rendszerét független élrendszernek nevezzük, ha a rendszerben szereplő élek páronként nem szomszédosak.

Képzeljük el, hogy egy ünnepség nyitótáncára több lány és fiú jelentkezett. A táncosok mindegyike nyilatkozott arról, hogy kit tart elfogadható partnernek a másik nem önkéntesei közül. A nyilatkozatok alapján meg lehet rajzolni egy gráfot, melynek csúcsai a nyitótáncra jelentkező személyek. Közéjük élet akkor húzunk, ha kölcsönösen elfogadják egymást táncpartnernek. Világos, hogy az ilymódon adódó gráf páros, és ha az alapján kijelölünk néhány táncoló párt, akkor ők egy független élrendszert generálnak a gráfban. Az ünnepség szervezői szeretnék, hogy a lehető legtöbb pár vegyen részt a nyitótáncon. Tehát egy maximális független élrendszert kellene kiválasztani az adott páros gráfból.

A megoldás alapötlete König Dénes magyar matematikus nevéhez fűződik és magyar módszernek vagy alternáló utak módszerének nevezik. Az eljárás feltételez egy kiindulási független élrendszert, amelyek elemszámát lépésenként eggyel növelve jut el az optimumhoz. Több megoldás is előfordulhat (pl. a kezdő független élrendszer megválasztása befolyásolhatja a végeredményt), de ezek egyenrangúak abban az értelemben, hogy a kiválasztott élek maximális száma ugyanannyi. A feladat általában egyetlen maximális független élrendszer megadása.

Kiindulási független élrendszert meghatározni nagyon egyszerű. Ennek szemléltetésére legyen ábrázolva a páros gráf úgy, hogy a $C_1 \subseteq C$ csúcsait egy felső, a $C_2 \subseteq C$ csúcsait egy alsó sorban helyezzük el, majd húzzuk be a két halmaz között a létező éleket (ld. 2.6. ábra).