11. Utazó ügynök heurisztikák

A közelítő eljárások létjogosultsága, hasznossága többféleképp indokolható. Az NP-nehéz problémák esetén nem ismeretesek polinomkorlátos műveletigényű megoldó eljárások. A rendelkezésre álló algoritmusok műveletigénye általában exponenciális függvénye a probléma méretének. Ennek következtében a nagyobb méretű feladatok megoldásának időigénye annyira nagy, hogy esetenként a megoldás nem realizálható. Így az NP-nehéz problémák esetében létjogosultsága van közelítő eljárásoknak. Ezek nem a tényleges optimális megoldást szolgáltatják, hanem egy lehetséges megoldást eredményeznek csak. Az így előállított lehetséges megoldással szemben alapvető elvárás, hogy "jó" legyen abban az értelemben, miszerint a hozzátartozó célfüggvényérték közel legyen az optimumértékhez. Az ilyen típusú eljárásokat szokásos *heurisztikus eljárásoknak* vagy egyszerűen *heurisztikáknak* nevezni.

A korábbiakban megismerkedtünk a $B\&B$ eljárással, amely igen sok NP-nehéz probléma esetében egy nagy műveletigényű algoritmus felépítését teszi lehetővé. Ezen eljárásokban a hatékonyság javítására felhasználhatók a rendelkezésre álló lehetséges megoldások. Minél jobb lehetséges megoldással rendelkezünk, általában annál gyorsabb, hatékonyabb lesz a $B\&B$ eljárás. Következésképp fontos az egyes problémacsoportokra olyan heurisztikák kidolgozása, amelyek gyorsak (polinomkorlátos műveletigényűek) és olyan lehetséges megoldást szolgáltatnak, melynek célfüggvényértéke közel van az optimumértékhez.

Az elmondottak kapcsán rögtön felvetődik a kérdés, hogy egy adott heurisztikát miként lehet minősíteni. Mikor mondhatjuk, hogy valamely heurisztika jó? Az elmúlt évtizedekben a heurisztikák vizsgálatára az alábbi három módszer alakult ki:

(1) legrosszabb esetek vizsgálata,

(2) valószínűségi analízis,

(3) empirikus analízis.

A következőkben rövid áttekintést adunk ezen módszerekről.

Legrosszabb esetek vizsgálata

Tekintsünk egy $\mathcal{C}$ minimalizálási problémaosztályt és jelöljön $\mathbf{A}$ egy, a tekintett problémák közelítő megoldását szolgáltató heurisztikát. A problémaosztály tetszőleges P problémájára jelölje $\mathbf{A}(P)$ a heurisztika által szolgáltatott lehetséges megoldáson felvett célfüggvényértéket és $\mathrm{OPT}(P)$ az optimumértéket. Ekkor az $\mathbf{A}$ algoritmust egy c konstansra c - *approximációs* algoritmusnak nevezzük, ha minden P probléma esetén $\mathbf{A}(P) \leq c \cdot \mathrm{OPT}(P)$. A legkisebb c értéket, amelyre az algoritmus c - approximációs az algoritmus *approximációs hányadosának* nevezzük. Az approximációs hányados egy további változata az *aszimptotikus approximációs hányados*. Az algoritmusra minden pozitív egész n-re definiáljuk az $R_n(\mathbf{A}) = \sup\{\mathbf{A}(P)/\mathrm{OPT}(P) : P \in \mathcal{C}, \mathrm{OPT}(P) \geq n\}$ értéket, és amennyiben létezik az $M_{\mathbf{A}} = \mathrm{limsup} R_n(\mathbf{A})$ érték, az adja meg a *heurisztika aszimptotikus approximációs hányadosát*. Az aszimptotikus approximációs hányados és az approximációs hányados közötti lényeges különbség az, hogy az aszimptotikus hányados azon inputokra, ahol az optimális költség kicsi nem követeli meg, hogy a heurisztika által adott megoldás az optimumhoz közeli érték legyen.

Az approximációs hányados illetve az aszimptotikus approximációs hányados nagysága a különböző problémaosztályokra és a különböző heurisztikákra más és más. Igen jó, 1 és 2 közé eső aszimptotikus approximációs hányadosok adódtak számos bin-packing heurisztikára. Ezzel szemben az általános utazó ügynök problémára 1976-ban Sahni és Gonzalez [**?**] igazolták, hogy amennyiben létezik konstans approximációs hányados valamely polinomkorlátos TSP-heurisztikára, akkor P=NP.

Az említett negatív eredmény alapján akár el is vethetnénk a polinomkorlátos TSP heurisztikák gondolatát. Azonban más problémaosztályoknál nyert tapasztalatok alapján a legrosszabb esetek vizsgálatának eredménye és az algoritmus gyakorlatban történő viselkedése között lényeges eltérés lehet. Erre szemléletes példa a szimplex algoritmus. Az igen sok megoldásra került lineáris programozási feladat alapján az átlagos iterációs lépésszám közelítőleg $3n$, ahol n a feladat egyenleteinek számát jelöli. Másrészt ismert olyan lineáris programozási feladat (ld. [3]), amelyre az iterációs lépésszám 2^n. További ilyen jellegű észrevételek azt támasztják alá, hogy az approximációs hányados bár lényeges információt ad az algoritmusról, de ennek alapján nem itélhető meg maradéktalanul az algoritmus jósága.

További érv a TSP heurisztikák vizsgálatához, hogy bizonyos speciális TSP feladatosztályokra léteznek viszonylag jó approximációs hányadossal rendelkező TSP heurisztikák .

Valószínűségi analízis

Ezekben a vizsgálatokban rögzített feladatméret mellett olyan P feladatot tekintenek, amelyben a paraméterek (együtthatók), mint független valószínűségi változók vannak megadva. Ekkor $\mathbf{A}(P)$ és $OPT(P)$, valamint hányadosuk is valószínűségi változó, melynek várható értékéből egy átlagos eltérésre lehet következtetni. Az így előállított mutatóval szemben kifogásolható, hogy a paramétereket megadó valószínűségi változók eloszlásaira tett feltételek nagyban befolyásolják azt, ugyanakkor a feltételek nem feltétlenül a problémaosztály jellemző tulajdonságait tükrözik. (Általában az egyenletes eloszlást teszik fel a feladatban szereplő valószínűségi változókra.)

Empirikus analízis

Az ilyen típusú vizsgálatokban konkrét problémamegoldásokból nyert eredmények alapján lehet bizonyos mutatókat képezni. A számítástechnika fejlődése, a gyors, nagyteljesítményű számítógépek megjelenése lehetővé tette azonos típusú nagyszámú probléma megoldását. Ezt kihasználva, rögzített számtartományból valamilyen (általában egyenletes) eloszlás mellett véletlenszerűen választva együtthatókat, majd az előállított konkrét problémáknak meghatározva az optimális, valamint a heurisztikus megoldását, képezhető a két célfüggvényérték hányadosa. Ezt elég sokszor ismételve, és átlagolva a hányadosokat, egyfajta empirikus mérőszámot kapunk. A vázolt vizsgálatban megkérdőjelezhető a választott eloszlás és számtartomány.

A fenti vázlatos áttekintésből kitűnik, hogy a heurisztikák jóságának vizsgálata igen bonyolult és csak részben megvalósítható feladat. Általában a heurisztikus eljárásokról az érdeklődők további meggondolásokat és részleteket találhatnak a [**?**] és [**?**] könyvekben. Mi a továbbiakban az egyszerűbb TSP heurisztikák közül fogunk néhányat ismertetni.

Az elmúlt időszakban számos TSP heurisztika került kidolgozásra és jelenleg is több kutató foglalkozik további eljárások felépítésén, illetve a meglévő eljárások vizsgálatával, javításával (ld. pl. [**?**], [2], [5], [**?**]). A különböző algoritmusok alapvetően három csoportba sorolhatók:

(1) körútépítő eljárások,

(2) körút javítását szolgáló algoritmusok,

(3) kombinált eljárások ((1) és (2) kombinációi).

A továbbiakban ismertetésre kerülő heurisztikák egy eljárás kivételével az (1) csoportba tartoznak. A tárgyalás egyszerűsítésének érdekében jelölje N az $\{1, \ldots, n\}$ halmazt.

Nearest addition

Előkészítő rész. Legyen $r = 1$, $I_r = \{1\}$, $E_r = \{(1, 1)\}$. (Az 1 város lesz az induló részkörút.) Térjünk rá az iterációs eljárásrészre.

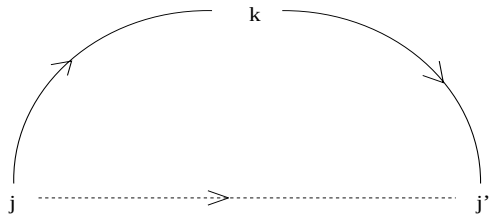
Iterációs rész (r. iteráció)

Ha $r = n$, akkor vége az eljárásnak, az E_r-beli élekből álló körút az eljárással szolgáltatott lehetséges megoldás. Ellenkező esetben határozzunk meg egy olyan $j \in I_r$, $k \in N \setminus I_r$ indexpárt, amelyre

$$c_{jk} = \min_{t \in N \setminus I_r} \{\min\{c_{st} : s \in I_r\}\} .$$

Legyen $I_{r+1} = I_r \cup \{k\}$. Mivel $j \in I_r$, ezért pontosan egy olyan $j' \in I_r$ index van, amelyre $(j, j') \in E_r$ ($r = 1$ esetén $j = j'$). Ekkor legyen $E_{r+1} = (E_r \setminus \{(j, j')\}) \cup \{(j, k), (k, j')\}$. Növeljük r értékét 1-gyel, és térjünk rá a következő iterációs lépésre.

Az eljárás úgy szemléltethető, hogy rendelkezésünkre áll egy részkörút ($r = 1$ esetén egyelemű részkörút). Ehhez kiválasztjuk a legközelebbi körúton kivüli várost. (Itt a legközelebbi város úgy értendő, hogy vesszük a körút pontjaiból a körúton kivüli pontokba vezető összes élet és ezen élek súlyainak a minimumát, ez a c_{jk} mennyiség.) Az így kiválasztott várossal bővítjük a körutat úgy, hogy a hozzá legközelebb eső, a körútban szereplő j városból ebbe a városba látogatunk, majd innen a j város j' leszármazottjába. Az alábbi ábra mutatja a beszúrási technikát. Az aktuális részkörútból töröljük a (j, j') élet és bővítjük a $(j, k), (k, j')$ élekkel.



11.1. ábra. Beszúrási technika.

A számításokat nagyban megkönnyíti egy olyan távolságvektor használata, amely minden iterációs lépésben megadja az aktuális részkörúton kivüli városoknak a részkörúttól való távolságát. Az első iterációs lépésben ez a $\mathbf{C}$ mátrix első sorvektora az $(1,1)$ indexű elem kivételével. Egyszerűen belátható, hogy amennyiben $\mathbf{v}_r$ az r-edik iterációs lépésben a távolságvektor, és ebben a lépésben a k várossal bővítjük a részkörutat, akkor $\mathbf{v}_{r+1}$-et megkapjuk úgy, hogy a körúton kívüli városokra rendre képezzük a $\mathbf{v}_r$ vektor megfelelő komponensének és a $\mathbf{C}$ mátrix k-adik sorában lévő megfelelő célfüggvényegyütthatónak a minimumát.

Az eljárás demonstrálására alkalmazzuk a fenti heurisztikát az előzőekben vizsgált 8-városos feladatra. Ekkor $\mathbf{v}_1 = (-, 2, 11, 10, 8, 7, 6, 5)$. Így az aktuális részkörúthoz (jelenleg az 1 várost tartalmazó részkörút) legközelebb levő város a 2-vel jelzett. Bővítve 2-vel a részkörutat, az $(1,2),(2,1)$ élekből álló új részkörúthoz jutunk. A távolságvektort a következőképpen tudjuk aktualizálni. A $\mathbf{v}_1$ vektor j-edik komponensének és c_{2j}-nek kell képezni a minimumát, ez lesz a $\mathbf{v}_2$ vektor j-edik komponense, ahol $j = 3, \ldots, 8$. A teljes eljárás végrehajtását az alábbi táblázatban foglaltuk össze, ahol az aktuális részkörutat ciklikus permutációként adjuk meg.

r	$\mathbf{v}_r$	(j,k)	részkörút
1	$(-, 2^*, 11, 10, 8, 7, 6, 5)$	$(1,2)$	$(1,2)$
2	$(-, -, 1^*, 8, 8, 4, 6, 5)$	$(2,3)$	$(1,2,3)$
3	$(-, -, -, 8, 8, 4, 3^*, 5)$	$(3,7)$	$(1,2,3,7)$
4	$(-, -, -, 8, 8, 4, -, 3^*)$	$(7,8)$	$(1,2,3,7,8)$
5	$(-, -, -, 8, 6, 3^*, -, -)$	$(8,6)$	$(1,2,3,7,8,6)$
6	$(-, -, -, 2^*, 6, -, -, -)$	$(6,4)$	$(1,2,3,7,8,6,4)$
7	$(-, -, -, -, 1^*, -, -, -)$	$(4,5)$	$(1,2,3,7,8,6,4,5)$

A tekintett feladatra az eljárás most pontosan egy optimális megoldást eredményezett. Ez egy kicsit megtévesztő. A vizsgált heurisztika nem ilyen jó. Az előzőekből tudjuk, hogy az általános TSP-re a $P \neq NP$ feltétel mellett nem létezik konstans approximációs hányados polinomkorlátos heurisztikák esetében. A tárgyalt eljárásról egyszerűen belátható, hogy polinomkorlátos, mégpedig $o(n^2)$ műveletigénnyel, így általános esetben nem várható hozzá kosntans approximációs hányados.

Nearest insertion

Az eljárás ugyanúgy épül fel, mint az előző. Az eltérés annyi, hogy nem a kiválasztott éllel bővítjük a részkörutat, hanem a kiválasztott, k-val jelölt városra meghatározzuk k lehető legjobb (legkisebb költséggel járó) beszúrását a részkörútba, és ezt hajtjuk végre.

Előkészítő rész. Legyen $r = 1$, $I_r = \{1\}$, $E_r = \{(1,1)\}$. Térjünk rá az iterációs eljárásrészre.

Iterációs rész (r. iteráció)

Ha $r = n$, akkor vége az eljárásnak, az E_r-beli élekből álló körút a heurisztikával előállított lehetséges megoldás. Ellenkező esetben határozzunk meg egy olyan $j \in I_r$, $k \in N \setminus I_r$ indexpárt, amelyre

$$c_{jk} = \min_{t \in N \setminus I_r} \{\min\{c_{st} : s \in I_r\}\} .$$

Legyen $I_{r+1} = I_r \cup \{k\}$, majd válasszunk egy olyan $(u,v) \in E_r$ élet, amelyre

$$\delta_{uv} = c_{uk} + c_{kv} - c_{uv} = \min\{c_{sk} + c_{kt} - c_{st} : (s,t) \in E_r\}.$$

Legyen $E_{r+1} = (E_r \setminus \{(u,v)\}) \cup \{(u,k),(k,v)\}$. Növeljük r értékét 1-gyel, és folytassuk az eljárást a következő iterációs lépéssel.

Ezen eljárásnál is használható az előzőekben bevezetett távolságvektor. A heurisztika bemutatására ismételten a 8-városos feladatot fogjuk használni. A végrehajtás egyes lépéseit a következő táblázat tartalmazza.

r	$\mathbf{v}_r$	k	(u,v)	δ_{uv}	részkörút
1	$(-, 2^*, 11, 10, 8, 7, 6, 5)$	2	$(1,1)$	-	$(1,2)$
2	$(-, -, 1^*, 8, 8, 4, 6, 5)$	3	$(2,1)$	0	$(1,2,3)$
3	$(-, -, -, 8, 8, 4, 3^*, 5)$	7	$(3,1)$	8	$(1,2,3,7)$
4	$(-, -, -, 8, 8, 4, -, 3^*)$	8	$(7,1)$	0	$(1,2,3,7,8)$
5	$(-, -, -, 8, 6, 3^*, -, -)$	6	$(2,3)$	8	$(1,2,6,3,7,8)$
6	$(-, -, -, 2^*, 6, -, -, -)$	4	$(6,3)$	7	$(1,2,6,4,3,7,8)$
7	$(-, -, -, -, 1^*, -, -, -)$	5	$(4,3)$	0	$(1,2,6,4,5,3,7,8)$

A kapott körúthoz tartozó célfüggvényérték 31, az optimum értéke 26, így a hányados 1.19. Az eljárással kapcsolatban megemlítjük, hogy műveletigénye $o(n^2)$, továbbá olyan speciális TSP feladatokra, melyek költségmátrixá-

ra teljesül a háromszögegyenlőtlenség és a költségmátrix szimmetrikus igazolást nyert (ld. [**?**]), hogy ebben az esetben az approximációs hányados 2.

Farthest insertion

Az eljárás analóg az előzőhöz. Az eltérés csupán annyi, hogy az aktuális részkörúttól legtávolabb levő várost választjuk ki, és ezt szúrjuk be a lehető legkisebb költséggel a részkörútba.

Itt is használható a távolságvektor. A heurisztikát ismét a 8-városos problémára alkalmazzuk, és a végrehajtás lépéseinek eredményét az alábbi táblázat tartalmazza.

r	$\mathbf{v}_r$	k	(u,v)	δ_{uv}	részkörút
1	$(-,2,11^*,10,8,7,6,5)$	3	$(1,1)$	-	$(1,3)$
2	$(-,2,-,10^*,8,7,3,5)$	4	$(1,3)$	9	$(1,4,3)$
3	$(-,2,-,-,1,7^*,3,5)$	6	$(1,4)$	-1	$(1,6,4,3)$
4	$(-,2,-,-,1,-,3,5^*)$	8	$(1,6)$	1	$(1,8,6,4,3)$
5	$(-,2^*,-,-,1,-,1,-)$	2	$(4,3)$	0	$(1,8,6,4,2,3)$
6	$(-,-,-,-,1^*,-,1,-)$	5	$(4,2)$	3	$(1,8,6,4,5,2,3)$
7	$(-,-,-,-,-,-,1,-)$	7	$(1,8)$	4	$(1,7,8,6,4,5,2,3)$

A kapott körúthoz tartozó célfüggvényérték 32.

Cheapest insertion

Hasonló az előzőekhez. A lényeges eltérés a részkörút bővítésére szolgáló város kiválasztásában van. Minden, az aktuális részkörúton kívüli k ($\in N \setminus I_r$) városra kiszámítjuk, hogy mennyi az aktuális részkörútba történő beszúrásának minimális költsége figyelembe véve az összes lehetséges beszúrásokat. Így minden $k \in N \setminus I_r$ városra adódik egy beszúrási költség. Ezek közül választunk egy minimálisat, és az ennek megfelelő város (minimális költségű) beszúrásával bővítjük a körutat.

A heurisztika végrehajtását a vizsgált példán szemléltetjük.

r	k	(u,v)	δ_{uv}	részkörút
1	2	$(1,1)$	-	$(1,2)$
2	3	$(2,1)$	0	$(1,2,3)$
3	6	$(2,3)$	8	$(1,2,6,3)$
4	5	$(2,6)$	6	$(1,2,5,6,3)$
5	4	$(2,5)$	1	$(1,2,4,5,6,3)$
6	7	$(2,4)$	8	$(1,2,7,4,5,6,3)$
7	8	$(2,7)$	2	$(1,2,8,7,4,5,6,3)$

A kapott körúthoz tartozó célfüggvényérték 33. Az eljárás műveletigénye $o(n^3)$. A [**?**] dolgozatban igazolást nyert, hogy a már említett speciális TSP-k esetében (szimmetrikus költségmátrixúak, amelyekre teljesül a háromszögegyenlőtlenség) az eljárás approxmációs hányadosa 2.

Tekintettel arra, hogy az ismertetett eljárások mindegyike igen gyors, továbbá az előállított körút függ a kiinduló várostól (mi ezt rendre 1-nek választottuk), lehetséges ugyanazt a heurisztikát többször is végrehajtani más és más kiindulási városokkal, majd az előálló körutak közül venni a legjobbat.

A tárgyalt heurisztikákat illetően egymástól függetlenül Adrabinski és Syslo [**?**], valamint Golden és társai [**?**] végeztek empirikus vizsgálatokat. A kapott eredmények azt mutatják, hogy a farthest insertion eljárás jobb értékeket szolgáltat általában, mint a másik három algoritmus. A következő táblázatban szereplő számsorokat a [**?**] munkából gyűjtöttük ki. Az adatok 5 darab 100×100-as euklidesi TSP-re vonatkoznak. (A városok egy euklidesi tér valamely síkjában vannak és távolságuk a térben a pontok távolsága.) A táblázatban azt adjuk meg, hogy a közelítő megoldás célfüggvényértéke hány százaléka az optimum értékének.

Nearest insertion (all cities)	118.69	117.81	122.96	114.44	120.33
Farthest insertion (all cities)	105.14	106.97	103.17	101.99	107.42
Cheapest insertion (all cities)	112.07	111.67	120.83	112.97	112.16

A következő eredményeket Adrabinski és Syslo [**?**] munkájából gyűjtöttük ki. (A feladatok méretei rendre $n = 20, 27, 42, 57, 120$).

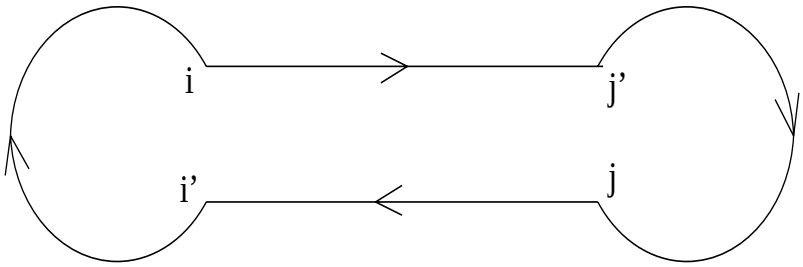
Nearest addition (one city)	186.58	113.88	111.02	114.76	121.78
Nearest insertion (one city)	138.21	106.89	110.87	109.25	117.27
Farthest insertion (one city)	128.45	100.35	101.43	101.28	103.06

Patching eljárások

A címben szereplő heurisztikák a hozzárendelési feladat és az utazó ügynök problémák kapcsolatára épülnek, és a következő észrevételen alapulnak. Oldjuk meg a megfelelő hozzárendelési feladatot. Ha az optimális megoldás körút, akkor megkaptuk a TSP optimális megoldását. Ellenkező esetben az optimális hozzárendelésnek megfelelő gráf diszjunkt részkörutak egyesítése. Ezen részkörutak számáról bizonyítást nyert, hogy amennyiben a költségmátrix elemeit egyenletes eloszlás alapján generáljuk, akkor a részkörútak számának várható értéke $\log(n)$. Így a részkörutak kis költséggel járó összekapcsolása, összefűzése egy jó heurisztikus megoldást eredményezhet. Attól függően, hogy miként kapcsoljuk össze a részkörutakat, különböző algoritmusok építhetők fel. A továbbiakban két ilyen típusú eljárást fogunk ismertetni.

2-patching eljárás

Tekintsünk két részkörutat. Jelölje I, J az egyes részkörutakban szereplő városok halmazát, továbbá tetszőleges k városra jelölje k' a k leszármazottját a k-t tartalmazó részkörútban. Legyen $i \in I$ és $j \in J$. Akkor törölve a részkörutakból az (i, i'), (j, j') éleket, és felvéve az (i, j'), (j, i') éleket, olyan részkörutat kapunk, amely az $I \cup J$-beli városokat tartalmazza. Az alábbi ábra mutatja az összekapcsolást.



11.2. ábra. A két részkörút összevonása.

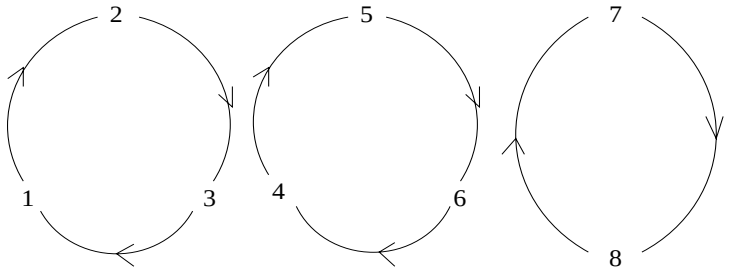
A két kör összefűzésével keletkező költséget egyszerűen megkaphatjuk, ez $c_{ij'} + c_{ji'} - c_{ii'} - c_{jj'}$. Véve ezeket a költségeket az összes $i \in I$ és $j \in J$ indexpárra, majd képezve ezek minimumát, megkapjuk a két részkör ilyen típusú "összekapcsolására" vonatkozó minimális költséget. Ezt *2-patching költségnek* és a részkörutak megfelelő összekapcsolását *2-patching műveletnek* nevezzük. Ezek után felépíthető az alábbi eljárás, amelyet R. M. Karp publikált 1979-ben a [**?**] dolgozatban.

Eljárás ([**?**])

Előkészítő rész. Oldjuk meg az $A(\mathbf{C})$ hozzárendelési feladatot. Ha $A(\mathbf{C})$ optimális megoldása körút, akkor vége az eljárásnak. Ellenkező esetben térjünk rá az iterációs eljárásrészre.

Iterációs rész. Válasszunk ki két legkisebb városszámú részkörutat és kapcsoljuk össze őket egy alkalmas 2-patching művelettel. Ha az előálló új megoldás (hozzárendelés) körút, akkor vége az eljárásnak. Ellenkező esetben térjünk rá a következő iterációra.

Az eljárás demonstrálására tekintsük ismét az előző példát. A hozzárendelési feladat optimális megoldása a következő három részkörútból áll.

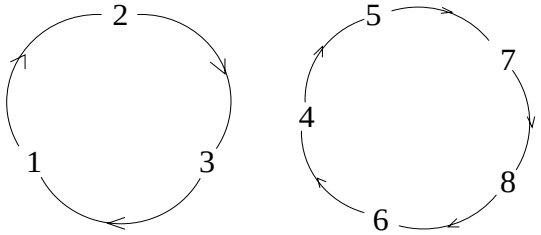


11.3. ábra. A kapott három részkörút.

Válasszuk ki a második és harmadik részkörutat. A kiválasztott részkörutakra a következő táblázatban adjuk meg a 2-patching költségeket. (A költségek számításánál az $\mathbf{A}^{(0)}$ mátrixot használtuk.)

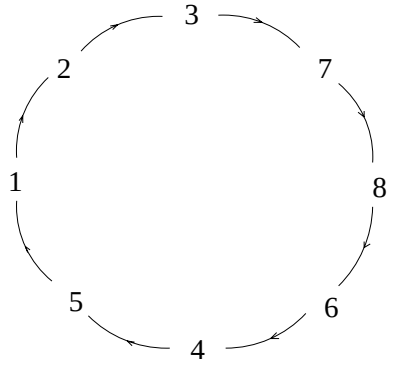
i	j	költség
4	7	$c^{(0)}_{48} + c^{(0)}_{75} - c^{(0)}_{45} - c^{(0)}_{78} = 9 + 6 = 15$
4	8	$c^{(0)}_{47} + c^{(0)}_{85} - c^{(0)}_{45} - c^{(0)}_{87} = 7 + 5 = 12$
5	7	$c^{(0)}_{58} + c^{(0)}_{76} - c^{(0)}_{56} - c^{(0)}_{78} = 7 + 9 = 16$
5	8	$c^{(0)}_{57} + c^{(0)}_{86} - c^{(0)}_{56} - c^{(0)}_{87} = 8 + 2 = 10^*$
6	7	$c^{(0)}_{68} + c^{(0)}_{74} - c^{(0)}_{64} - c^{(0)}_{78} = 7 + 7 = 14$
6	8	$c^{(0)}_{67} + c^{(0)}_{84} - c^{(0)}_{64} - c^{(0)}_{87} = 9 + 9 = 18$

A 2-patching költség 10. A megfelelő 2-patching művelet során az $(5, 7)$ és $(8, 6)$ éleket kell felvenni és az $(5, 6)$, $(8, 7)$ éleket kell törölni. Az új megoldás a 11.4. ábrán megadott két részkörútból áll:



11.4. ábra. Részkörutak az összevonás után.

Kiszámítva a fenti két körre a 2-patching költséget, az $i = 3$, $j = 5$ esetben kapjuk a minimumot. Törölve a $(3, 1)$, $(5, 7)$ éleket, valamint felvéve a $(3, 7)$, $(5, 1)$ éleket, az alábbi körutat kapjuk, amely éppen egy optimális megoldás.



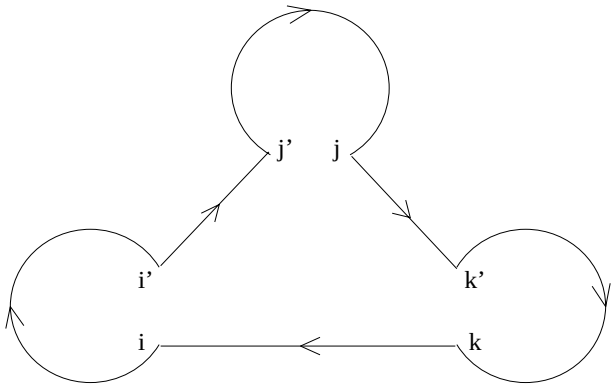
11.5. ábra. A 2-patching eljárással nyert optimális megoldás.

Az ismertetett 2-patching eljárásra J. M. Steele és R. M. Karp végzett valószínűségi analízist (ld. [**?**]). Igazolták, hogy a [0,1] intervallumon egyenletes eloszlású független valószínűségi változók esetén a közelítő érték és az optimumérték hányadosának várható értéke $1 + o(n^{-\frac{1}{2}})$.

3-patching eljárás

Kettőnél több kör összekapcsolása hatékonyabb eljárást eredményezhet. Ezen az észrevételen alapul a [**?**] dolgozatban közölt *3-patching eljárás*, melynek megadásához szükségesek bizonyos előkészületek.

Tekintsünk három részkörutat. Jelölje rendre I, J, K a részkörutakban szereplő városok halmazát. Legyen $i \in I$, $j \in J$, $k \in K$, továbbá jelölje i', j', k' rendre az i, j, k leszármazottait az I, J, K halmazokhoz tartozó részkörutakban. Akkor törölve az (i, i'), (j, j'), (k, k') éleket, és felvéve az (i, j'), (j, k'), (k, i') éleket, a három részkörutat egyetlen részkörútba (körútba) tudjuk egyesíteni. A 11.6. ábra szemlélteti a három részkörút egyesítését.



11.6. ábra. Három részkörút összekapcsolása.

A három részkörút tekintett összekapcsolásának költsége:

$$c_{ij'} + c_{jk'} + c_{ki'} - c_{ii'} - c_{jj'} - c_{kk'} \ .$$

Rendre meghatározva ezeket a költségeket minden $i \in I$, $j \in J$, $k \in K$ indexre, megkapjuk a három részkörút ilyen típusú összekapcsolására vonatkozó minimális költséget. Ezt a költséget *3-patching költségnek*, és a részkörutak ennek megfelelő összekapcsolását *3-patching műveletnek* nevezzük.

Ezek után felépíthető a következő heurisztikus eljárás.

Eljárás

Előkészítő rész. Oldjuk meg az $A(\mathbf{C})$ hozzárendelési feladatot. Ha $A(\mathbf{C})$ optimális megoldása körút, akkor vége az eljárásnak. Ellenkező esetben térjünk rá az iterációs eljárásrészre.

Iterációs rész

- *1. lépés.* Jelölje m az aktuális hozzárendelés részkörútjainak a számát. Ha $m \leq 9$, akkor a 2. lépés következik. Ellenkező esetben rendezzük a részkörutakat elemszámuk tágabb értelemben vett növekvő sorrendjébe. Jelölje a rendezett sorozatot $U_1, \ldots, U_m$. Számítsuk ki a d_{rs} 2-patching költségeket az U_r, U_{m-l+s} részkörutakra minden $1 \leq r \leq l$, $1 \leq s \leq l$ indexpárra, ahol $l = [m/2]$. Oldjuk meg a (d_{ij}) költségmátrixú $l \times l$-es hozzárendelési feladatot, és jelölje az optimális hozzárendelést φ. Hajtsunk végre egy $d_{r\varphi(r)}$ költségű 2-patching műveletet az $U_r, U_{m-l+\varphi(r)}$ részkörutakon minden $1 \leq r \leq l$ indexre. Az így képezett részkörutak által meghatározott hozzárendelést tekintve aktuális hozzárendelésnek, folytassuk az eljárást az 1. lépés ismétlésével.

- *2. lépés.* Ha az aktuális hozzárendelés körút, akkor vége az eljárásnak. Ellenkező esetben, ha $m < 3$, akkor hajtsunk végre a két részkörúton egy 2-patching műveletet, és vége az eljárásnak. $m \geq 3$ esetén határozzunk meg a részkörutak közül három olyan részkörutat, amelyek 3-patching költsége minimális. Hajtsunk végre a kiválasztott három részkörúton egy, a minimális költségnek megfelelő 3-patching műveletet. Az új hozzárendelést tekintve aktuális hozzárendelésnek, és részkörútjainak számát az aktuális m-nek, folytassuk az eljárást a 2. lépés ismétlésével.

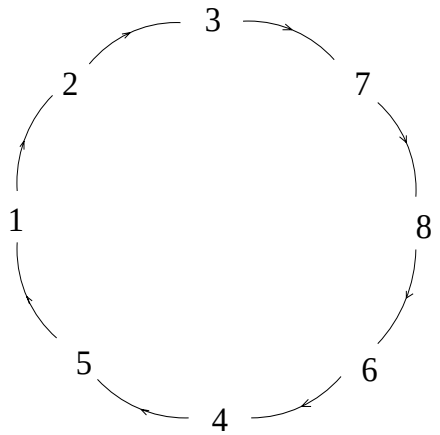
A fenti eljárásban $m > 9$ esetén a kis köröket párosítjuk a nagy körökkel, és olyan párosításra törekszünk, hogy az összepárosított köröket összefűzve, a költségek növekedése ne legyen jelentős. A 9 konstans használatát az indokolja, hogy fixpont nélküli permutációk esetében a részkörök számának várható értéke közelítőleg $\log(n)$, ahol n a városok számát jelöli. Végül az eljárás hatékony végrehajtásához bevezethető egy 3-dimenziós tömb, amely

a részkörutakra a 3-patching költségeket tartalmazza. Az első ilyen tömb alapján a további tömbök már viszonylag kis műveletigénnyel kiszámíthatók.

Az eljárás szemléltetésére tekintsük ismét a korábbiakban vizsgált 8-városos feladatot. A megfelelő hozzárendelési feladat optimális megoldása az $(1,2),(2,3),(3,1)$, a $(4,5),(5,6),(6,4)$ és a $(7,8),(8,7)$ éleket tartalmazó részkörutakból áll. A $c_{ij'} + c_{jk'} + c_{ki'} - c_{ii'} - c_{jj'} - c_{kk'}$ költségeket Δ_{ijk}-val jelölve, a konkrét értékeket a következő táblázatban adtuk meg:

i	j	k	Δ_{ijk}	i	j	k	Δ_{ijk}	i	j	k	Δ_{ijk}	i	j	k	Δ_{ijk}
1	4	7	23	2	5	8	20	1	7	4	17	2	8	4	19
1	4	8	22	2	6	7	23	1	7	5	21	2	8	5	14
1	5	7	20	2	6	8	25	1	7	6	16	2	8	6	17
1	5	8	22	3	4	7	19	1	8	4	17	3	7	4	22
1	6	7	23	3	4	8	16	1	8	5	15	3	7	5	24
1	6	8	26	3	5	7	21	1	8	6	19	3	7	6	23
2	4	7	25	3	5	8	21	2	7	4	21	3	8	4	13
2	4	8	23	3	6	7	20	2	7	5	22	3	8	5	9^*
2	5	7	19	3	6	8	21	2	7	6	16	3	8	6	17

Az $i = 3$, $j = 8$ és $k = 5$ indexekre kapjuk a legkisebb értéket. Végrehajtva a megfelelő 3-patching műveletet, az alábbi körút adódik, amely speciálisan optimális megoldás is.



11.7. ábra. A 3-patching eljárással nyert optimális megoldás.

Az ismertetett eljárásokkal kapcsolatosan a [**?**] munkában egy empirikus analízis található, amelyben ezen eljárások nyertek összehasonlítást. A vizsgálatok során rendre minden tekintett méret mellett 100-100 probléma került generálásra. A célfüggvényegyütthatók a $0, 1, \ldots, 100$ egészek közül egyenletes eloszlás mellett lettek generálva. A számítógépes vizsgálat eredményét a következő táblázatban foglaltuk össze. Négy városszámra, az $n = 100$, $n = 150$, $n = 200$ és $n = 250$ városszámokra történt a 100-100 utazó ügynök probléma generálása. A táblázat baloldala tartalmazza a vizsgált eljárásokat, és a megfelelő sor adja meg az illető eljáráshoz tartozó értékeket. Minden városméretre az első oszlop adja meg az érintett eljárással meghatározott célfüggvényértékek valamint az optimumértékek hányadosainak átlagát, a második oszlop tartalmazza az érintett eljárás futási idejeinek átlagát, végül a harmadik oszlop azt mutatja, hogy az érintett eljárás hány esetben szolgáltatta a legjobb célfüggvényértéket. Ez a B&B eljárásnál üres maradt, mivel ez mindig az optimális megoldást szolgáltatja.

	$n = 100$			$n = 150$			$n = 200$			$n = 250$		
	average ratio	average sec.	best value	average ratio	average sec.	best value	average ratio	average sec.	best value	average ratio	average sec.	best value
B&B	1.000	40.78	-	1.000	87.69	-	1.000	194.1	-	1.000	320.9	-
3-patching k=5	1.054	19.53	88	1.056	58.55	81	1.052	190.2	82	1.059	370.8	80
3-patching k=3	1.061	11.60	70	1.063	34.93	67	1.061	122.0	54	1.078	218.6	48
3-patching k=1	1.069	3.84	55	1.096	11.90	39	1.094	39.8	25	1.134	72.6	24
2-patching k=5	1.090	11.14	33	1.082	29.70	38	1.069	88.4	40	1.101	158.3	37
2-patching k=3	1.092	6.69	28	1.097	17.92	26	1.085	53.1	27	1.119	94.8	23
2-patching k=1	1.108	2.21	21	1.127	6.04	15	1.127	17.7	13	1.177	31.5	12
cheapest insertion	4.654	7.77	0	6.794	37.48	0	9.934	89.7	0	18.11	175.4	0
nearest insertion	4.392	9.19	0	7.047	43.66	0	11.39	104.6	0	18.60	205.1	0
farthest insertion	4.534	8.76	0	7.110	43.71	0	11.39	104.6	0	18.60	205.3	0
nearest addition	18.43	7.09	0	33.84	32.72	0	57.51	77.0	0	98.43	149.7	0

A fenti táblázat szépen mutatja, a következő tendenciákat. Valamennyi patching eljárás egészen jó szuboptimális megoldásokat szolgáltat, és ez a "jóság" nem változik szignifikánsan a feladatok méretének növekedésével.

Ezzel szemben a különböző beszúrási eljárásokra azt tapasztalhatjuk, hogy az általuk meghatározott lehetséges megoldások célfüggvényértéke többszöröse az optimumértéknek es ez szignifikánsan romlik a feladatok méreteinek növekedésével.

Az eddig bemutatott heurisztikus eljárások mindegyike az általános esetre (a költségmátrixról nem tételezünk fel semmit) lett kifejlesztve. Ettől függetlenül ezek az eljárások alkalmazhatók olyan TSP feladatosztályokra, ahol a költségmátrixokra különböző megszorításokat teszünk. A bemutatott eljárásokhoz fűzött megjegyzések azt mutatják, hogy bizonyos megszorítások mellett a heurisztikák viselkedéséről többet tudunk állítani.

A továbbiakban speciális TSP feladatosztályokat vizsgálunk és ezekre kifejlesztett heurisztikus eljárásokat mutatunk be. Elsőként a szimmetrikus esetet vizsgáljuk ($c_{ij} = c_{ji}$, $i = 1, \ldots, n; j = 1, \ldots, n$), ami megfelel az irányítatlan gráfok esetének, és erre adunk meg egy körútjavító heurisztikus eljárást.

Az ismertetésre kerülő eljárás azon az észrevételen alapul, hogy amennyiben adott egy körút, úgy abból törölve két nem szomszédos élet, a körút két diszjunkt útra esik szét. Ezek után létezik két olyan egyértelműen meghatározott él, hogy ezekkel bővítve a két útból álló gráfot, az eredmény egy másik körút lesz. (Például, ha az $(1,2), \ldots (4,5), (1,5)$ élekből álló körútból töröljük a $(2,3)$ és $(4,5)$ éleket, akkor az előálló két útból csak a $(2,4)$ és $(3,5)$ élek felvételével készíthető körút. A továbbiak egyszerűsítésének érdekében nevezzünk az $\bar{\mathbf{X}}$ *körút szomszédjának* minden olyan körutat, amely előáll $\bar{\mathbf{X}}$-ből két él törlésével, és két új él felvételével. Egyszerűen belátható, hogy $\bar{\mathbf{X}}^{n\times n}$ szomszédjainak a száma $n(n-3)/2$, ha önmagát nem számítjuk szomszédnak.

2-optimális eljárás

Előkészítő rész. Határozzuk meg valamilyen eljárással a feladat egy $\bar{\mathbf{X}}$ körútját. Legyen $\mathbf{X}^{(0)} = \bar{\mathbf{X}}$, $r = 0$, és térjünk rá az iterációs részre.

Iterációs rész (r. iteráció)

- *1. lépés.* Határozzuk meg $\mathbf{X}^{(r)}$ összes szomszédját. Ha $\mathbf{X}^{(r)}$ minden $\bar{\mathbf{X}}$ szomszédjára $z(\mathbf{X}^{(r)}) \leq z(\bar{\mathbf{X}})$ teljesül, akkor vége az eljárásnak, $\mathbf{X}^{(r)}$ az eljárással előállított körút. Ellenkező esetben a 2. lépés következik.

- *2. lépés.* Jelöljön $\bar{\mathbf{X}}$ $\mathbf{X}^{(r)}$ szomszédjai közül egy olyan körutat, amelyen a z függvény a szomszédokra vonatkozóan minimális értéket vesz fel. Legyen $\mathbf{X}^{(r+1)} = \bar{\mathbf{X}}$, növeljük r értékét eggyel, és térjünk rá a következő iterációs lépésre.

Az eljárás bemutatására tekintsük a következő példát.

11.1. példa. Tekintsük azt az 5 városos utazó ügynök feladatot, amelynek költségmátrixa az alábbi $\mathbf{C}$ mátrix és legyen az induló körútunk az a körút, amely az

$$\mathbf{X}^{(0)} : 1 - -2 - -3 - -4 - -5 - -1$$

sorrendben megy végig a városokon. Ekkor $z(\mathbf{X}^{(0)}) = 20$.

$$\mathbf{C} = \begin{pmatrix} & 3 & 4 & 1 & 2 \\ & & 2 & 5 & 3 \\ & & & 6 & 2 \\ & & & & 7 \\ & & & & \end{pmatrix}$$

A kiindulási körút szomszédjait tartalmazza a következő táblázat.

r	törölt élek	$\mathbf{X}^{(r)}$ szomszédja	$z(\bar{\mathbf{X}})$
0	$(1,2),(3,4)$	$1-3-2-4-5-1$	20
0	$(1,2),(4,5)$	$1-4-3-2-5-1$	14
0	$(2,3),(4,5)$	$1-2-4-3-5-1$	18
0	$(2,3),(1,5)$	$1-2-5-4-3-1$	23
0	$(3,4),(1,5)$	$1-2-3-5-4-1$	15

Tehát

$$\mathbf{X}^{(1)} : 1 - -4 - -3 - -2 - -5 - -1, \;\; z(\mathbf{X}^{(1)}) = 14.$$

Az eljárás következő iterációs lépésében számolt szonszédokat tartalmazza a következő táblázat.

r	törölt élek	$\mathbf{X}^{(r)}$ szomszédja	$z(\bar{\mathbf{X}})$
1	$(1,4),(2,3)$	$1-3-4-2-5-1$	20
1	$(1,4),(2,5)$	$1-2-3-4-5-1$	20
1	$(3,4),(2,5)$	$1-4-2-3-5-1$	12
1	$(1,5),(3,4)$	$1-3-2-5-4-1$	17
1	$(2,3),(1,5)$	$1-4-3-5-2-1$	15

Ekkor

$$\mathbf{X}^{(2)} : 1 - -4 - -2 - -3 - -5 - -1, \;\; z(\mathbf{X}^{(2)}) = 12.$$

Az eljárás által a következő iterációs lépésben számolt szonszédokat tartalmazza a következő táblázat.

r	törölt élek	$\mathbf{X}^{(r)}$ szomszédja	$z(\bar{\mathbf{X}})$
2	$(1,4),(2,3)$	$1-2-4-3-5-1$	18
2	$(1,4),(3,5)$	$1-3-2-4-5-1$	20
2	$(2,4),(3,5)$	$1-4-3-2-5-1$	14
2	$(2,4),(1,5)$	$1-4-5-3-2-1$	15
2	$(2,3),(1,5)$	$1-4-2-5-3-1$	15

Ekkor $\mathbf{X}^{(2)}$ minden $\bar{\mathbf{X}}$ szomszédjára $z(\mathbf{X}^{(2)}) \leq z(\bar{\mathbf{X}})$ teljesül, így vége az eljárásnak, és a heurisztika által kapott megoldás az $(1,4,2,3,5,1)$ körút.

A fejezet hátralevő részében olyan TSP problémákat vizsgálunk, amelyekben a költségmátrix szimmetrikus és teljesül a háromszögegyenlőtlenség. Az ilyen mátrixok esetére ismertetünk egy 3/2-approximációs heurisztikát.

Az eljárás ismertetéséhez a probléma gráfelméleti reprezentációját használjuk. A problémához egy teljes (bármely két pont között megy él) irányítatlan hálózatot rendelünk, amelynek csúcsai a városoknak felelnek meg, és az éleken a hosszak a két várost összekötő út hosszai. A feladat ekkor az, hogy megtaláljuk azt az összes ponton átmenő kört, amelyre a körben szereplő élek hosszainak összege minimális.

Az eljárás megadásához szükségünk lesz a következő definícióra. Egy Euler féle multigráf Euler körének a *rövidítésén* pontoknak azt a sorozatát értjük, amelyet úgy kapunk, hogy a körben minden csúcsnak csak az első előfordulását tartjuk meg, amennyiben többször is szerepel a további előfordulásait töröljük a pontok listájából.

11.2. példa A 2.5. példában tekintett multigráfra meghatározott Euler kör pontok sorozatával a következőképpen írható le: $(1,6,5,4,3,2,5,2,1)$. Az Euler kör rövidítése az $(1,6,5,4,3,2)$ pontsorozatot adja meg.

Egy Euler kör rövidítése pontok egy sorbarendezését adja meg, amelyből egy körutat kapunk, ha a kezdőpontot összekötjük a végponttal. Amennyiben a pontok multigráfjához tartozó élekre teljesül a háromszögegyenlőtlenség, akkor az Euler körben szereplő élek súlyainak összegére és a rövidítéshez tartozó körútban szereplő élek hosszainak összegére teljesül a következő állítás.

11.1. segédtétel *A rövidítéshez tartozó körútban az élek súlyainak összege legfeljebb annyi, mint az Euler körben az élek súlyainak összege.*

Bizonyítás. Legyen $(p_1,\ldots,p_n)$ a rövidítésben a pontok sorozata. A továbbiakban a p_{n+1} pontot a p_1 pontnak feleltetjük meg. Jelölje az Euler

körben a p_i és p_{i+1} pontok első előfordulásai közötti pontokat $q_{i1}, \ldots, q_{ik_i}$, ahol a pontok ezen sorozata üres is lehet.

A háromszög egyenlőtlenséget alkalmazva többször egymás után adódik, hogy minden i-re teljesül $c_{p_ip_{i+1}} \leq c_{p_iq_{i1}} + \sum_{j=1}^{k_i-1} c_{q_{ij},q_{i,j+1}} + c_{q_{ik_i}p_{i+1}}$. Ezen értékeket összegezve adódik a segédtétel állítása.

A Euler körök rövidítésének alkalmazásával megadhatjuk Christofides heurisztikus algoritmusát.

Christofides eljárása ([1])

- *1. lépés.* Határozzunk meg a feladatot leíró $\mathcal{G}$ hálózatban egy minimális feszítőfát Kruskal algoritmusa alapján. Jelölje ezt a fát T.
- *2. lépés.* Legyen $\mathcal{G}'$ az a részgráfja $\mathcal{G}$-nek, amelynek a pontjai T páratlan fokszámú pontjai, és az élei $\mathcal{G}$ azon élei, amelyek ezen pontok között mennek. Határozzuk meg a 3.3. fejezetben leírt algoritmus alapján $\mathcal{G}'$ egy minimális költségű teljes párosítását. Egészítsük ki a párosításban szereplő élekkel a T fát.
- *3. lépés.* Az így kapott multigráfban határozzunk meg egy Euler kört.
- *4. lépés.* Vegyük a kapott Euler kör rövidítését, a rövidítéshez tartozó körút a heurisztikával előállított megoldás.

Elsőként igazoljuk, hogy az eljárás végrehajtható.

11.2. segédtétel *A Christofides eljárásban megadott lépések mindegyike végrehajtható, és az eredmény egy körút lesz.*

Bizonyítás. Mivel a kiindulási gráf egy teljes gráf, ezért összefüggő, így Kruskal algoritmusa valóban egy feszítőfát ad eredményül. Mivel egy gráfban a csúcsok fokszámainak összege az élek számának kétszerese (minden élt beszámolunk mindkét végpontjánál), ezért a fokszámok összege páros. Másrészt ha a fokszámok összege páros, akkor páros azon pontoknak a száma, amelyeknek páratlan a fokszáma. Így a 2. lépésben tekintett $\mathcal{G}$ gráfnak páros számú csúcsa van, továbbá a gráf teljes, így a párosítási feladatnak van optimális megoldása. A megoldásban szereplő éleket hozzávéve T-hez egy olyan összefüggő multigráfhoz jutunk, amelyben minden csúcs fokszáma páros (azon pontok fokszáma, amelyeké páratlan volt 1-el növekszik). Tehát a multigráf kielégíti a 2.1. tétel feltételeit, így van Euler köre, tehát a 3. lépés végrehajtható. Mivel az Euler kör rövidítése során valóban

egy körútat kapunk, azért a 4. lépés után egy lehetséges megoldáshoz jutunk. Fontos megjegyeznünk, hogy mivel minden lépés polinomiális időben végrehajtható, ezért a teljes algoritmus is.

Az algoritmus egy optimálishoz közeli megoldást ad, miként azt a következő állítás mutatja.

11.1. tétel *Christofides heurisztikus algoritmusa* 3/2*-approximációs algoritmus.*

Bizonyítás Tekintsünk egy tetszőleges P TSP problémát. Jelölje az eljárás első lépése során kapott T feszítőfa hosszát $c(T)$. Jelölje a második lépésben megkonstruált párosításra a párosításban szereplő élek összhosszát $c(M)$. Végül jelöljük a probléma optimális megoldásának a költségét OPT(P)-vel.

Elsőként vegyük észre, hogy egy körútból elhagyva bármely élét egy feszítőfát kapunk, így a minimális feszítőfa hossza kisebb, mint az optimális körút költsége, azaz $c(T) \leq OPT(P)$.

Most vizsgáljuk a párosítás költségét. Legyenek $p_1, p_2, \ldots, p_{2j}$ a T feszítőfa páratlan fokszámú pontjai. Az általánosság megszorítása nélkül feltehetjük, hogy ezek a pontok ilyen sorrendben helyezkednek el a P probléma optimális megoldást adó körútjában is, hiszen ezt a pontok átindexelésével elérhetjük. Másrészt ekkor a 11.1 segédtétel bizonyításához hasonlóan a háromszögegyenlőtlenség többszöri alkalmazásával azt kapjuk, hogy a $c_{p_i,p_{i+1}}$ távolság legfeljebb annyi mint az optimális körútban a p_i és a p_{i+1} pontokat összekötő útszakaszban az élek hosszainak összege. Következésképpen $\sum_{i=1}^{2j-1} c_{p_i,p_{i+1}} + c_{p_{2j},p_1} \leq OPT(P)$.

Vegyük a következő párosításait a $p_1, p_2, \ldots, p_{2j}$ pontoknak. Legyen M_1 a (p_n, p_1) és a (p_{2i}, p_{2i+1}), $(i = 1, \ldots, j-1)$ pontpárokat összekötő élek halmaza, M_2 pedig a (p_{2i+1}, p_{2i+2}), $(i = 0, \ldots, j-1)$ pontpárokat összekötő élek halmaza. Ekkor a két párosítás költségének az összege $\sum_{i=1}^{2j-1} c_{p_i,p_{i+1}} + c_{p_{2j},p_1} \leq OPT(P)$. Ezen észrevétel alapján adódik, hogy a két párosítás közül arra, amelynek kisebb a költsége, ez a költség legfeljebb $OPT(P)/2$, másrészt ebből következik, hogy $c(M) \leq OPT(P)$.

A $c(T)$ és $c(M)$ értékekre adott becsléseink alapján adódik, hogy az eljárás 3. lépésében megkonstruált Euler körben az élek hosszainak az összege legfeljebb $\frac{3}{2}OPT(P)$, amiből a 11.1. segédtétel alapján adódik, hogy a heurisztika által szolgáltatott megoldásban is legfeljebb $\frac{3}{2}OPT(P)$ az élek hosszainak az összege, amivel a tétel állítását igazoltuk.

A fejezet lezárásként a következő példán mutatjuk be Christofides algoritmusát.

11.3. példa. Tekintsük azt az 5 városos utazó ügynök feladatot, amelynek költségmátrixa az alábbi $\mathbf{C}$ mátrix

$$\mathbf{C} = \begin{pmatrix} & 2 & 3 & 3 & 5 \\ & & 4 & 4 & 2 \\ & & & 3 & 3 \\ & & & & 6 \\ & & & & \end{pmatrix}$$

Elsőként hajtsuk végre Kruskal algoritmusát. Az algoritmus a következő éleket választja ki a megadott sorrendben $(1,2),(2,5),(1,3),(1,4)$. A kapott feszítőfa hossza 10. A feszítőfában a páratlan fokszámú pontok $1,3,4,5$. Megoldva az ezen pontokból és a közöttük futó élekből álló gráfra a párosítási problémát, azt kapjuk, hogy a minimális költségű teljes párosítás az $(1,4)$ és $(3,5)$ éleket tartalmazza, és a párosítás költség 6. Tehát a 3. lépésben vizsgált multigráfnak az élei $(1,2),(1,3),(1,4),(1,4),(2,5),(3,5)$. Az Euler kört kereső algoritmus a következő pontsorozatot adja vissza $1,2,5,3,1,4,1$. Ezen Euler körnek a rövidítése az $1,2,5,3,4$ sorrendje a pontoknak, így a Christofides algoritmus által adott körút az $1-2-5-3-4-1$, amely körútnak a költsége 13.

Bibliography

[1] Christofides, N. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical report, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh PA, 1976.

[2] Johnson, D. S., L. A. McGeoch, The traveling salesman problem: A case study in local optimization, in: *Local Search in Combinatorial Optimization*, (eds.: E. H. L. , Aarts, J. K. Lenstra), Wiley, New York, 1997, 215-310.

[3] Klee, V., G. J. Minty, How good is the the simplex algorithm?, in: *Inequalities*, III, (ed.: O. Shisha), Academic Press, New York, 1972, 159-175.

[4] Lin, S., Computer Solutions of the Traveling Salesman Problem, *Bell Syst. Tech. J.* **44** (1965), 2245-2269.

[5] Reinelt, G., *The Traveling Salesman: Computational Solutions for TSP Applications*, Springer-Verlag, Berlin, 1994.