Eötvös Loránd Tudományegyetem

Természettudományi Kar

Krendl Eszter

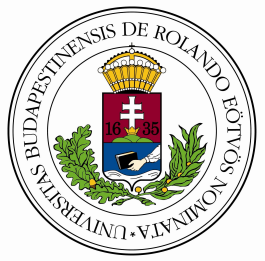
Reptéri transzferek gráfelméleti megközelítése

BSc Szakdolgozat

Témavezető:

Frank András
emeritus professzor

Operációkutatás Tanszék



Budapest, 2022

Köszönetnyilvánítás

Szeretném megköszönni témavezetőmnek, Frank Andrásnak felém mutatott türelmét és támogatását valamint érdeklődését a témámmal kapcsolatban.

Tartalomjegyzék

Bevezetés

Szakdolgozatom témáját egy valós üzleti probléma ihlette, mellyel munkám során találkoztam. A cégünk feladata egy olyan algoritmus megalkotása volt, ami optimális vagy közel optimális megoldást ad arra, hogy különböző helyen, különböző utasokat kell felvenni és elszállítani egy közös úti célhoz vagy fordítva egy közös kiindulóponton felvenni őket és különböző helyekre elszállítani, hazavinni őket.

Ehhez hasonló feladatok több különböző iparágban is felmerülnek különböző narratívával. Hasonló problémakörbe tartozik az utazóügynök probléma, ahol egy úttal kell minden pontot pontosan egyszer érinteni és a jármű útvonaltervezési probléma is, ahol nem utasokat, hanem különböző méretű, súlyú csomagok szállítása a feladat. Ezek mindegyike gráfként ábrázolható, így optimalizációs gráfalgoritmusokat keresünk megoldásként.

A különböző feladatok megközelíthetőek több nézőpontból, több megoldási úton is. Megoldások lehetnek heurisztikus vagy közelítő algoritmusok.

Én azt választottam, hogy a gráfok particionálásának szemszögéből fogom megvizsgálni a fent vázolt problémakört. Egy megfelelően particionált gráf optimális megoldást nyújthat arra is, hogy mely utasokat érdemes egy autóba beültetni, kiket érdemes együtt utaztatni.

1. fejezet

Problémafelvetés

Ebben a fejezetben a szakdolgozatom kiinduló üzleti problémájáról fogok vázolni, illetve az üzleti problémát megpróbálom átfogalmazni, átfordítani a matematika nyelvére és így a későbbiekben már a matematikai feladattal, problémával foglalkozom.

1.1. Üzleti probléma

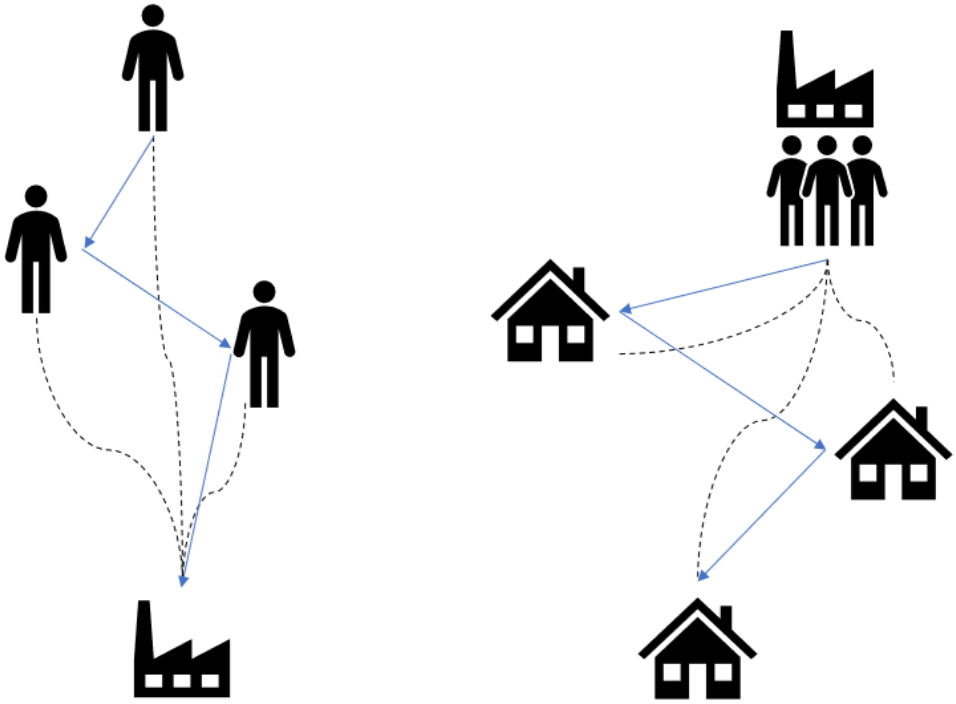
Az üzleti probléma vázolásához képeljük el, hogy egy fuvarszervező cégnél dolgozunk (köznyelven taxis cégnél), amely szerződésben áll több, nagyobb partnerrel is, amelyek velünk, mint fuvarszervező cég szállítatják az utasaikat. Az egyik partnerünk azzal keres meg minket, hogy szeretné a hajnali órákban - amikor nincs tömegközlekedés - a mi szolgáltatásunkat igénybe venni, hogy az alkalmazottai bejussanak a reptérre dolgozni vagy hazajussanak onnan munka után. Azonban minden egyes munkavállalót egyesével szállítani rendkívül költséges lenne, ezért olyan megoldást szeretne, amiben egy taxi több alkalmazottat szállít egyszerre.

Ehhez értelemszerűen azokat a munkavállalókat érdemes egy autóba ültetetni, akik közel laknak egymáshoz vagy egy útvonalon laknak.

A cég még további igényeket is megfogalmaz, ami az utasok kényelmét szolgálja, az a kérése, hogy egy utas ne utazzon túl sokáig, vagyis ne üljön egy óránál tovább a taxiban. Ezen felül maga a taxi is egy természetes fizikai korlátot alkot, mivel egy normál taxiba nem fér be 3 embernél több.

Cél

Üzleti feladat az utasokat úgy összeültetni, hogy az a legköltséghatékonyabb legyen a fizikai és kényelmi határok figyelembevételével.



1.2. Matematikai megfogalmazás

Adott n+1 pont a síkban (utasok kiindulópontok és a reptér, mint közös érkezési vagy indulási pont). Ezek a pontok egy gráf csúcsait határozzák meg. A gráf élei irányítottak és súlyozottak, az n "utascsúcs" között mindkét irányba megy él és minden pontból megy egy irányított él a kiemelt "reptércsúcsba". A gráf élsúlyai többféleképp is meghatározhatóak:

•Két pont közötti időbeli távolság közúton,

•Két pont közötti térbeli távolság közúton,

•Fenti két érték egy tetszőleges függvénye (mint fuvardíj).

A gráfot akár irányítatlanul is fel lehet rajzolni, mivel az esetek nagy részében meg fog egyezni az ellentétes irányokon az élsúly.

Feladat

Utak olyan halmazának a megkeresése, amik uniójára a követezők teljesülnek:

•a reptércsúcs kivételével minden csúcsot pontosan egyszer lefed,

•végpontja a "reptércsúcsban" van,

•és minimális lesz az összköltségük.

Az utakra a következő peremfeltételeknek kell teljesülni:

•Egy úton maximum 3+1 csúcs helyezkedhet el (3 utas és a reptér).

•Egy úton eltöltött idő nem több, mint egy óra (kivéve ha csak egy utascsúcsot fed le).

A feladat akár meg is fordítható, azaz közös kiindulópontból az egyes utasok hazaszállítása is tekinthető. A fenti feltételek a megfordított feladat kapcsán is ugyanazok.

Másképp megfogalmazva olyan csúcs diszjunkt, maximum 3+1 csúcsot összekötő utakat keresünk, melyek lefedik az összes csúcsot, nem hosszabbak egy óránál és összköltségük minimális.

1.3. Egy triviális megoldás

A fenti feladat egy megoldása lehet az, hogy az összes lehetséges megoldást megkeressük és kiválasztjuk belőle az optimális, a legkisebb költséggel rendelkezőt.

Az összes megoldás egy megkeresésének egy módja kétkörösre vehető. Az első körben sorba rendezzük az utasokat az összes lehetséges módon (mintha egy jármű menne mindegyikért), majd az algoritmus második felében ezeken a sorbamegyünk és maximum 3 fős csoportokra bontjuk (annak megfelelően, hogy a peremfeltételek teljesüljenek).

Könnyen belátható, hogy ez nem egy járható út, mert kis elemszámra is hamar elszáll az algoritmus.

Az elemeket sorb állítani n! féleképp lehet. Majd ezeből ismétlés nélküli variációval 3 fős csoportokat képezni $\frac{n!}{(n-3)!}$ féleképp lehet.

1.4. "Jó", de nem optimális megoldási algoritmus

Az alábbi algoritmus egy mohó algoritmussal "elég jó", de nem optimális megoldást kínál a fenti feladatra. Az algoritmus előnye, hogy polinomiális futási idővel rendelkezik.

Az algoritmus első lépése, hogy minden utascsúcsot közvetlenül összekötünk a reptércsúccsal. Ez a kiinduló élhalmazunk. Ezen keresünk javítóutakat.

Javító út keresésénk első lépése, hogy megkeressük azt a két pontot, ami a legközelebb esik egymáshoz. Ez az új út úgy nézne ki, hogy a két pontot összekötő él és az egyik pontból induló, reptérpontba induló él adná a javítást. Ez az út javító út, ha nem szegi meg a peremfeltételeket (maximum 3+1 csúcs lehet az úton és nem hosszabb egy óránál), valamint az új út rövidebb, mint az összevonás előtti utak összege.

Ha találtunk egy javító utat, akkor az eredeti utakat a javító úttal helyettesítjük és újabb javító utat keresünk a gráfban. Ha nem találunk több javító utat, akkor megáll az algoritmus és a talált utak adják ki a megoldást.

A valóságban cégünk ennek az algoritmusnak egy kissé módosított változatát használja, így a tapasztalat azt mutatja, hogy ez valóban az ügyfél számára egy elfogadható megoldást nyújt az utasok összeültetésére.

2. fejezet

Matematikai alapok, fogalmak

Ha a témakört matematikai úton akarjuk megközelíteni, akkor ennek alapját is be kell vezetünk. Ez a fejezet szolgálja ezt a célt. Ebben a fejezetben bevezetésre kerülnek a témához tartozó matematikai alapok, definíciók, tételek, algoritmusok, melyek a későbbiekben a megértést segítik, felhasználásra kerülnek. Az itt található definicíók nagy részét az irodalomjeegyzékben található [1], [2], [4], [6] számú kiadványok, valamit saját, még egyetemi éveim alatt kézzel írt jegyzeteim adják.

2.1. Gráfelméleti alapok

2.1.1. Definíció (Fenyő). *Egy irányított F=(S,E) fát s-fenyőnek nevezünk, ha a gráf minden pontja elérhető egyirányú úton s-ből.*

2.1.2. Definíció (Feszítő fenyő). *Egy F=(S,E') fenyő a D=(V,E) gráf feszítő fenyője, ha a csúcshalamzok megegyeznek (S=V) és F élei a D élei közül kerülnek ki úgy, hogy fenyőt kapjunk (E'⊂E).*

2.1.3. Definíció (Páros gráf). *Egy G=(V(A,B),E) gráf páros, ha G pontjainak V=(G) halmaza két részre, A-ra és B-re osztható úgy, hogy G minden élének egyik végpontja A-ban, másik végpontja B-ben van.*

2.1.4. Definíció (Merevkörű gráf (chordal graph)). *G=(V,E) merev körű gráf, ha minden négy vagy több csúcsot tartalmazó körének van húrja, azaz olyan éle, ami nem része a körnek, de összeköt két a körbe tartozó csúcsot.*

2.1.5. Definíció (Gyengén merevkörű gráf vagy páros húrgráf (chordal bipartite graph)). *G=(A,B;E) gyengén merev körű gráf, ha minden hat vagy több csúcsot tartalmazó körének van húrja, azaz olyan éle, ami nem része a körnek, de összeköt két a körbe tartozó csúcsot és emellett páros gráf.*

2.1.6. Definíció (Összehasonlítási gráf (comparability graph)). *G=(V,E) összehasonítási gráf. ha olyan irányítatlan gráf, melynek csúcsai egy részben rendezett halamaz elemeit reprezentálják, melyek között akkor van él behúzva, ha a csúcsokkal szimbolizált elemek részbenrendezéssel egymással összehasonlíthatóak.*

2.1.7. Definíció (Páros permutációs gráf (bipartite permutation graph)). *G=(V,E) gárf permutációs gráf, ha a csúcsai a permutáció elemeit reprezentálják és az élek azon elemek között futnak, melyek felcserélésre kerülnek a permutáció során. Egy G=(V,E) permutációs gráf, ha G és G komplementere is összehasonlítási gráf.*

2.1.8. Definíció (Kaktusz (cacti vagy cactus)). *G=(V,E) gráf kaktusz, ha nincs olyan éle, amit a gráfban egynél több kör tartalmaz.*

2.1.9. Definíció (Út-particionálás). *G=(V,E) gráf út particionálása a G csúcsdiszjunkt utainak halmaza.* $P_1 = \{V_1, E_1\}, \ldots P_r = \{V_r, E_r\}$, *amelyek uniója lefedi a G gráf V csúcshalmazát.*
Vagyis $V_i \cap V_j = \varnothing \ \forall i \neq j$*-re, és* $\bigcup_{i=1}^{r} V_i = V$

2.1.10. Definíció (k-út particionálás). *Olyan út-parcionálás, ahol minden* P_i *halmaz maximum k db csúcsot tartalmaz.*

2.1.11. Definíció (Hamilton-út). *A gráf minden csúcsát pontosan egyszer tartalmazó út.*

2.1.12. Definíció (Hamilton-kör). *A gráf minden csúcsát pontosan egyszer tartalmazó kör (a kiindulási- és végpontot leszámítva).*

2.1.13. Definíció (Párosítás). *Adott G gráf néhány független élének halmaza (azaz, olyan élek halmaza, melyeknek nincs közös csúcsa).*

2.1.1. Mélységi keresés

Ebben a fejezetben a gráf mélységi bejárásának algoritmusa kerül bemutatásra, mely egy gráf csúcshalamazának feltérképezésére, bejárására szolgál. A egy gráf mélységi bejárásánál egy csúcsból kiindulva, onnan egy úton haladva megkeressük a legtávolabbi pontot, amit el tudunk érni az adott úton, majd visszatérünk a kiindulástól legtávolabb eső elágazáshoz és onnan ismét megkeressük a legtávolabbi pontot és így tovább. Ha egy adott ágon már nincs több elágazásunk olyan csúcsba, amit már ne láttunk volna, akkor visszamegyünk a kiindulási pontba és egy újabb, még nem látott csúcson indulunk el. Az algoritmus végén a mélységi bejárással a gráf egy feszítőfáját kapjuk eredményül.

Az algoritmus leírása során a következő jelöléseket fogom használni, melyek megegyeznek az irodalomjegyzék [7] számú írásában található jelölésekkel. A lentebb vázolt algoritmus alapját is ez képezi.

- $d(v)$: a v csúcs mélységi száma, azaz, amikor az algoritmus először eléri az adott csúcsot
- $f(v)$: a v csúcs befejezési száma, azaz, amikor az algoritmus utoljára találkozik a csúccsal
- $m(v)$: a v-t megelőző csúcs (v őse vagy szülője), vagyis az a csúcs, amiből az algoritmus futása során elértük a v csúcsot
- a: a jelenlegi vizsgált csúcs
- s: algoritmus kiindulási csúcsa
- D: az eddigi legnagyobb mélységi szám
- F: az eddigi legnagyobb befejezési szám

Mélységi keresés algoritmusa

(1) Elős lépésben beállítjuk a kezdeti értékeket.

$d(s) := 1$, $d(v) := \infty$ minden $v \in V$ és $v \neq s$ esetén, $f(v) := \infty$ minden $v \in V$-re, $m(v) := \infty$ minden $v \in V$-re, $D := 1$, $F := 0$, $a := s$.

(2) Vizsgáljuk meg az a szomszédjait, keressünk egy v csúcsot, melyre $av \in E$

(2.1) Ha van olyan $av \in E$, amire $d(v) = \infty$, akkor $D := D + 1$, $d(v) = D$, $m(v) = a$. Vagyis, ha a-ból vezet él olyan csúcsba, amit még nem láttunk az algoritmus során, akkor a fa maximális mélységi számát növeljük meg és az újonnan talált v csúcsnak legyen ez a mélységi száma. Jelöljük meg a-t a v csúcs szülőcsúcsának és ezután a v csúcsra nézzük ugyanezt a vizsgálatot és visszamegyünk a (2) lépésre.

(2.2) Ha nincs ilyen csúcs, akkor $F := F + 1$, $f(a) := F$, azaz a legnagyobb befejezési számot növeljük meg eggyel és legyen ez a befejezési száma. Vagyis ebben a lépésben találkoztunk utoljára az a csúccsal, ezért ezt megjelöljük.

(2.2.1) Ha $m(a) \neq \infty$, akkor legyen $a := m(a)$, azaz ha ismerjük az a szülőcsúcsát (ez akkor lehet, ha még nem értünk vissza a kiindulócsúcsba), akkor ismét azt a csúcsot kezdjük el vizsgálni és visszamegyünk a (2) lépésre.

(2.2.2) Ha van másik olyan csúcs, melyre $d(v) = \infty$, akkor $a := v$, azaz, ha van még olyan csúcs, melyet még nem jártunk be, akkor ugorjunk arra a csúcsra és indítsuk el onnan ismét a mélységi bejárást. (Ha összefüggő, irányítatlan gráfunk van, akkor nem lesz ilyen csúcs.)

Ezután megáll az algoritmus, mert minden csúcsot bejártunk. Az algoritmus során minden csúcsnál megjelöltük az adott csúcs szülőjét, valamint azt is, hogy a kiindulócsúcstól milyen távol találtuk meg az adott csúcsot. Ezen kívül a kapott fa mélységét is eltároljuk.

3. fejezet

Hasonló problémák

3.1. Jármű úvonaltervezési probléma (Capacitated Vehicle Routing Problem[CVRP])

A jámű úvonaltervezési probléma esetén adott d_i mennyiség minden egyes áruhoz, melyeket n db vevőnek kell kiszállítani, ahol $i \in \{i \dots n\} = N$. Ezek egy központi depóból kerülnek kiszállításra (jelölje ezt a $\{0\}$ pont). A kiszállításhoz k darab, egyenkén C kapacitással bíró jármű szolgál. Két pont közötti költséget $c_{ij} \geq 0$ jelöli $(0 \leq i, j \leq n)$. A költségek szimmetrikusak, ha $c_{ij} = c_{ji}$ és $c_{ii} = 0$. Feladat az összes áru kiszállítása a költségek minimalizálása mellett.

Ez a kiinduló üzleti problémánk egy általánosítása. Ha az n vevőnek megfeleltetjük az n darab utasunkat, akiknél a mennyiség konstans 1 ($\forall i : d(i) = 1$, mivel minden utas egységnyi helyet foglal el a taxiban), a depónak a repteret, és az összes jármű kapacitás 3, akkor a fent vázolt üzleti problémával fog megegyezni (az időre vonatkozó peremfeltétlet figyelmen kívül hagyva).

Ebben a témakörben főként olyan anyagokat találtam, melyek heurisztikus algoritmusokat adnak, melyek "jósága" gyakorlati vizsgálaton, azaz az algoritmusok tényleges futtatásával vannak igazolva, azonban arról nem esik bennük szó, hogy ezen algoritmusoknak az eredménye mennyire esik messze az optimumtól.

Ezen oknál fogva szakdolgozatom további részében az ületi problémát nem ebből az irányból közelítem meg, hanem egy másik, a kiindulási problémához szintén hasonló problémakör felől közelítek.

3.2. Gráfok k-út particionálási problémája

Egy másik hasonló, gráfelméleti probléma a gráfok út particionálási problémája. Az út parcionálási problémánál adott egy $G = (V, E)$ gárf, melyben úgy keresünk minimális számú utakat, hogy az utak uniója pontosan egyszer fedje le a gráf összes csúcsát. Ezt a minimális út-számot $p(G)$-vel jelöljük. Könnyen belátható, hogy $p(G) = 1$ akkor és csak akkor, ha a gráfban létezik Hamilton-út. Azokban a típusú gráfokban, ahol a Hamilton-út keresése NP-nehéz feladatok közé tartozik, ott $p(G)$ meghatározása is NP-nehéz.

Tekintsük egy általánosítását az út particionálás problémának. Egy út particionálást k-út particionálásnak nevezünk, ha minden út maximum k csúcsot fed le, ahol k egy adott pozitív egész szám. A k-út particionálás problémája az utak minimális számának meghatározása k-út particionálással, ezt a minimális számot $p_k(G)$-vel jelöljük.

Amennyiben az üzleti problémánknál nem vesszük figyelembe az időre vonatkozó peremfeltételt, akkor $k = 4$ esetén a k-út particionálás a számunkra is optimális megoldást fogja adni, azaz kiderül, hogy hány taxira van szükség az utasok elszállításához, illetve egyes algoritmusok magukat a partíciókat is megadják.

Szakdolgozatom további részében főként a súlyozatlan esetre fogok koncentrálni, ami megegyezik azzal, mintha minden utas egységnyi távolságra lakna egymástól, ami sajnos kevésbé fogja fedni a valóságot, azaz a tényleges üzleti probléma helyett annak inkább az elméleti hátterével fogok foglalkozni.

4. fejezet

A k-út particionálás vizsgálata

Szakdolgozatom további részében a kiinduló problémát a k-út particionálás problémájának szemszögéből fogom vizsgálni. Mivel a k-út particionálás már $k \geq 3$ esetén is NP-teljes probléma általános gráfok esetén, ezért a témakörben található egyszerűsítéseket, azaz különböző, speciális gráfok k-út bontását, fogom vizsgálni az egyszerűbb gráfoktól a bonyolultabbakig haladva.

Először az egyetemi éveim alatt is megismert $k = 2$ eset fogom áttekinteni, azaz a teljes párosítás keresését. Ezután fákkal majd kaktuszokkal foglalkozok, melyek a gráfok egy speciális csoportját képezik és létezik rájuk olyan algoritmus, mely polinomiális időben megtalálja a $p_k(G)$-t, illetve megadja a particionálást is. Majd egy speciális páros gráfra, a páros permutációs gráfokra mutatom be a szintén polinomiális időben lefutó particionálási algoritmust.

4.1. k=2 eset, a teljes párosítás

Ha a k-út particionálás problémája esetén k=2-vel, akkor ez megegyezik a gráfban a teljes párosítás keresésével. Azaz minden út maximum két csúcsot fed le és a gráfot csúcsdiszjunkt utakkal kell lefedni. Ezzel a problémával már az egyetemi évek alatt találtkoztunk és létezik polinomális időben lefutó algoritmus is, mely megtalálja az optimális párosítást páros gráfok esetén. Szakdolgozatom teljes párosítással foglalkozó részéhez az irodalomjegyzék [1] és [4] szakirodalmait valamit egyetemi jegyzeteimet használtam fel.

Annak eldöntésére, hogy az gráfban létezik-e teljes párosítás az alábbi tételek szolgálnak:

4.1.1. Tétel (Hall-tétel). *A G=(A,B;E) páros gráfban akkor és csak akkor van teljes párosítás, ha* $|A| = |B|$ *és* $\forall X \subseteq A$*-ra* $|N(X)| \geq |X|$*, ahol N(X) az X beli csúcsok szomszédjait jelöli.*

4.1.2. Tétel (Tutte-tétel). *A G=(V,E) gráfban akkor és csak akkor van teljes párosítás, ha* $\forall X \subseteq V(G)$ *elhagyása esetén a keletkező páratlan komponensek száma* $\leq |X|$.

Maximális párosítás páros gráfban

Páros gráf esetén a Magyar módszert használva található meg a teljes párosítás. A módszer javítóutak keresésére épül. Ehhez vezessük be a következő fogalmakat és jelöléseket:

Legyen M egy tetszőleges párosítás a G gráfban.

4.1.3. Definíció. *1. A v csúcsot szabad csúcsnak nevezzük, ha $d_M(v) = 0$ (azaz, ha M-ben 0 a fokszáma), egyébként a v csúcsot fedett csúcsnak hívjuk.*

2. Egy út alternáló út, ha élei felváltva $\in M$ és $\notin M$.

3. Egy alternáló út javító út, ha mindkét vége szabad csúcs, azaz a párosítás által le nem fedett csúcs.

Az algoritmus során a célunk javító út keresése, mellyel a jelenleginél nagyobb párosítást kapunk. Javító út a szélességi keresés egy módosított verziójával megtalálható.

Maximális párosítás meghatározása - Kőnig algoritmusa

Az alábbi módszerrel egy $G = (S, T; E)$ páros gráfban keressük meg a maximális párosítást.

(1) Induljunk ki egy tetszőleges párosításból, legyen ez M (akár az üres halmazból is kiindulhatunk). Jelöljük S_M-mel az S csúcshalmaz halmaz azon csúcsait, melyeket az M párosítás lefed. Hasonlóan T_M-mel jelöljük a T csúcshalmaz halmaz azon csúcsait, melyeket az M párosítás lefed.

(2) Irányítsuk meg az M párosítás éleit T-ből S-be és a maradék csúcsokat $(E-M)$ pedig S-ből T-be.

(3) Keressünk egy olyan P utat, mely $S - S_M$-ből $T - T_M$-be megy, azaz két fedetlen csúcsot köt össze. Ez a P út egy javítóút lesz a gráfban. Ha nem találtunk ilyen élt, akkor ugorjunk az (5) lépésre.

(4) Ezen P út mentén fordítsuk meg az élek irányítását, és frissítsük az M párosításunkat és S_M, T_M halmazokat ennek megfelelően majd ugorjunk vissza a (2) lépésre.

(5) Az algoritmus véget ért és megtaláltuk a maximális párosítást a gráfban.

Magyar módszer algoritmusa

A magyar módszernél a kiindulópontunk egy $G = (S, T; E)$ páros gráf, melynek élein adott egy c költségfüggvény. Az algoritmus során egy maximális súlyú teljes párosítást keresünk. Ehhez vezessük be a következő jelöléseket:

Legyen az y a gráf egy súlyozott lefogása, ahol $y_u + y_v \geq c_{uv}$, ahol $uv \in E$. Egy él akkor pontos, ha egyenlőség áll fent, azaz $y_u + y_v = c_{uv}$. Adott y lefogó vektorra $G_y = (S, T; E_y)$, ahol $E_y = \{uv \in E : y_u + y_v = c_{uv}\}$.

(1) Az algoritmus során egy tetszőleges y-ból indulunk ki.

(2) Hajtsuk végre Kőnig algoritmusát a G_y gráfra. Ha megtaláljuk benne a teljes párosítást, akkor végeztünk, a további lépéseket nem hajtjuk végre.

(3) Ha az előző lépésben nem találtuk meg a teljes párosítást, akkor legyen M a maximális párosítás G_y-ban és D_y^M a G_y irányítása az M párosítás szerint. Z jelölje a D_y^M-ben az $S - S_M$-ből elérhető csúcsok halmazát.

(4) Ha D^M-ben nem lép ki él Z-ből, akkor G-ben nincs teljes párosítás és végeztünk.

(5) Legyen $\delta = \min\{y_u + y_v - c_{uv} : u \in S \cap Z, v \in T - Z, uv \in E\}$.

$$y = \begin{cases} y_u - \delta & \text{ha } u \in Z \cap S \\ y_u + \delta & \text{ha } u \in Z \cap T \\ y_u & \text{egyébként} \end{cases}$$

Ekkor y továbbra is lefogó vektor és $|Z|$ legalább eggyel nő. A fenti lépéseket ismételve eljutunk az optimális megoldásig.

4.2. Fák particionálása

Szakdolgozatom következő részében fákkal fogok foglalkozni. Célom az lesz, hogy egy polinomiális időben lefutó algoritmus segítségével megmondhassuk, hogy tetszőleges fában mennyi a minimális k-út particiók száma. Az algoritmus csak ezt a számot fogja megadni, de magukat a particiókat nem határozza meg. Fák particionálási számának meghatározásához az irodalomjegyzék [5] számú cikkében található tételeket és algoritmust használtam fel.

Ezekhez először különböző jelöléseket és elnevezéseket fogok bevezetni, melyeket a szakdolgozatom további részében felhasználok:

•$p_k(G)$: minimális száma a k-utaknak a G gráfban, azaz a G minimális k-út particiójának száma,

•$p_k(G, v)$: v gyökérrel rendelkező k-út partició, azaz a partició egyik útja a v csúcsban végződik,

•$l_k(G, v)$: a minimális csúcsszáma a v csúcsot tartalmazó útnak a v gyökércsúccsal rendelkező $p_k(G, v)$ k-út particionálásban.

A következőkben olyan állításokat és tételeket vezetek be, melyek az algoritmus megalkotásához lesznek szükségesek.

4.2.1. Állítás. *Legyen a $G = (V, E)$ gráf, melyben a v egy gyökér csúcs. Ekkor*

$$p_k(G, v) - 1 \leq p_k(G) \leq p_k(G, v). \tag{4.1}$$

Bizonyítás. A gyökércsúccsal rendelkező k-út parcionálás is egy a G-ben lévő k-út particionálások közül, ezért az egyenlőtlenség jobb oldala egyértelmű, $p_k(G) \leq p_k(G, v)$. Nézzük a bal oldali egyenlőtlenséget. Ehhez tegyük fel, hogy a G-ben az optimális k-út particiónk P, melyben p az az út, ami tartalmazza a v csúcsot. Válasszuk ketté a p utat a p_1-re és p_2-re úgy, hogy a v csúcs legyen a p_1 út utolsó csúcsa (előfordulhat, p_2 üres út lesz, ha az eredeti felosztásban is a v csúcsban végződött a p út). Ekkor a P felosztásban a p utat helyettesíthetjük a p_1 és p_2 utakkal, így G-nek egy v gyökerű particioját kapjuk, melynek mérete legfeljebb $p_k(G) + 1$ lesz. Ebből következik is az állítás bal oldalának egyenlőtlensége: $p_k(G, v) - 1 \leq p_k(G)$ □

4.2.2. Állítás. *Ha a G gráf felírható a G_1 és a G_2 gráf kompozíciójaként, amikben a v_1 és v_2 gyökércsúcsok, akkor a következőket mondhatjuk:*

$$p_k(G_1) + p_k(G_2) - 1 \leq p_k(G) \leq p_k(G_1) + p_k(G_2) \tag{4.2}$$

$$p_k(G_1, v_1) + p_k(G_2) - 1 \leq p_k(G) \leq p_k(G_1, v_1) + p_k(G_2) \tag{4.3}$$

Bizonyítás. 4.2 egyenlet bizonyításához tegyük fel, hogy P az optimális particionálása G-nek. Ekkor ez a partició felosztható P_{G_1} és P_{G_2} particionálásokra, ahol a P_{G_1} a P particionálás G_1 beli csúcsait és éleit tartalmazza, P_{G_2} hasonlóan. Ebből látszik, hogy $p_k(G) \geq |P_{G_1}| + |P_{G_2}| - 1 \geq p_k(G_1) + p_k(G_2) - 1$.

Az egyenlőtlengés jobb oldalának bizonyításához tegyük fel, hogy P_1 és P_2 a G_1 és G_2 részgráfok optimális particionálásai. Ekkor a két particionálás uniója szintén egy particionálást ad ki az egész, G gráfra, tehát $p_k(G) \leq p_k(G_1) + p_k(G_2)$ teljesül.

4.3 egyenlet bizonyítás hasonlóan megy a 4.2 rész bizonyításához, csak feltételezzük, hogy P és P_1 is v gyökércsúccsal rendelkező particionálások. □

4.2.3. Tétel. *Ha a G gráf felírható a G_1 és a G_2 gráf kompozíciójaként, amikben a v_1 és v_2 gyökércsúcsok, akkor a következőket mondhatjuk:*

$$p_k(G) = \begin{cases} p_k(G_1) + p_k(G_2) - 1 & \text{ha } |A| = 2 \text{ és } l_k(G_1, v_1) + l_k(G_2, v_2) \leq k, \\ p_k(G_1) + p_k(G_2) & \text{egyébként,} \end{cases} \tag{4.4}$$

ahol $|A| = \{i : p_k(G_i) = p_k(G_i, v_i),\ i = 1 \text{ vagy } 2\}$

Bizonyítás. A 4.2.2 tétel 4.2 egyenletéből láttuk, hogy $p_k(G_1)+p_k(G_2)-1 \leq p_k(G) \leq p_k(G_1)+p_k(G_2)$. Tehát a bizonyítás első részében csak azt bizonyítjuk, hogy $p_k(G) = p_k(G_1) + p_k(G_2)$ akkor és csak akkor, ha $|A| = 2$ és $l_k(g_1, v_1) + l_k(G_2, v_2) \leq k$. Tegyük fel, hogy igaz a következő egyenlőség: $p_k(G) = p_k(G_1)+p_k(G_2)$. Legyen P egy optimális particionálása G-nek és legyen q az az út benne, ami tartalmazza a v_1 csúcsot. Ha $q \cap V(G_2) = \emptyset$, akkor $|P_{G_1}|+|P_{G_2}| = p_k(G) = p_k(G_1)+p_k(G_2)-1$ (második egyenlőség a feltevésünk). Ebből az következne, hogy vagy $|P_{G_1}| < p_k(G_1)$ vagy $|P_{G_2}| < p_k(G_2)$, ami egy ellentmondás.

Tehát $q \cap V(G_2) \neq \emptyset$. P_{G_i} a G_i részgráf egy optimális,v_i gyökkér rendelkező k-út particionálása $i = 1, 2$-re. Mivel $|P_{G_1}|+|P_{G_2}| = p_k(G)+1 = p_k(G_1)+p_k(G_2)$ és $p_k(G_i) \leq p_k(G-i, v_i) \leq |P_{G_i}| = p_k(G_i)$ $(i = 1, 2)$ (az első egyenlőtlenség a 4.2.1 tételből következik, a második pedig abból, hogy P_{G_i} gyökérrel rendelkező particionálás volt), ezért $p_k(G_i) = p_k(G_i, v_i)$ $i = 1, 2$ esetén. Ebből következőleg $|A| = 2$ és $l_k(g_1, v_1) + l_k(G_2, v_2) \leq |q| \leq k$.

A másik irány bizonyításához tegyük fel, hogy $|A| = 2$ és $l_k(g_1, v_1)+l_k(G_2, v_2) \leq k$ teljesül. Legyen P_i egy optimális, gyökérrel rendlekező k-út particionálása a G_i gárfnak, ahol q_i út tartalmazza a v_i csúcsot $(i = 1, 2$-re). Mivel $|q_1|+|q_2| = l_k(g_1, v_1)+l_k(G_2, v_2) \leq k$, ezért a $(P_1 \cup P_2 \cup \{q_1 \cup \{v_1v_2\} \cup q_2\}) - \{q_1, q_2\}$ egy k-út particionálása a G gráfnak, melynek a mérete $p_k(G_1, v_1)+p_k(G_2, v_2)-1 = p_k(G_1)+p_k(G_2)-1$ (mert feltettük, hogy $|A| = 2$), tehát $p_k(G_1) + p_k(G_2) - 1 = p_k(G)$. □

4.2.4. Tétel. *Ha a G gráf felírható a G_1 és a G_2 gráf kompozíciójaként, amikben a v_1 és v_2 gyökércsúcsok, akkor a következőket mondhatjuk:*

$$p_k(G, v_1) = \begin{cases} p_k(G_1, v_1) + p_k(G_2) - 1 & \text{ha } l_k(G_1, v_1) = 1 \text{ és } l_k(G_2, v_2) < k, \text{ és } p_k(G_2) = p_k(G_2, v_2) \\ p_k(G_1, v_1) + p_k(G_2) & \text{egyébként.} \end{cases} \tag{4.5}$$

Bizonyítás. A 4.2.2 tétel 4.3 egyenletéből láttuk, hogy $p_k(G_1, v_1)+p_k(G_2)-1 \leq p_k(G, v_1) \leq p_k(G_1, v_1)+p_k(G_2)$. Ezert csak annak a bizonyítása van hátra, hogy $p_k(G_1, v_1)+p_k(G_2)-1 = p_k(G, v_1)$ akkor és csak akkor, ha $l_k(G_1, v_1) = 1$ és $l_k(G_2, v_2) < k$, és $p_k(G_2) = p_k(G_2, v_2)$.

Először tegyük fel, hogy $p_k(G_1, v_1) + p_k(G_2) - 1 = p_k(G, v_1)$. Legyen P egy optimális k-út particionálása a G gárfnak, amiben q az az út, ami tartalmazza a v_1 csúcsot. Legyenek P_{G_1} és P_{G_2} a G_1 és G_2 gárfok gyökércsúccsal rendelkező k-út particionálásai. Ha feltesszük, hogy $q \cap v(G_2) = \emptyset$, akkor $|P_{G_1}| + |P_{G_2}| = p_k(G, v_1) = p_k(G_1, v_1) + p_k(G_2) - 1$ Az első egyenlőséget azért feltételezük, mert ha a q útnak nincs közös pontja a G_2 gráffal, akkor a két komponens particionálása megegyezne a teljes gráf particionálásával. A második egyenlőség pedig a kiinduló feltevésünkből jön. Azonban ez az egyenlőség ellentmondásra vezet, mert vagy $P_{G_1} < p_k(G_1)$ vagy $P_{G_2} < p_k(G_2)$, ami nem lehet, mert mind a kettő 1-1 particionálás és ezek mérete nem lehet kisebb az optimális particionálás méreténél.

Ezért feltehetjük, hogy $q \cap v(G_2) \neq \emptyset$ és hogy $q \cap v(G_1) = \{v_1\}$. Ebből következik, hogy a P_{G_2} szintén egy gyökércsúccsal rendelkező k-útparticionálása G_2-nek. Mivel $|P_{G_1}| + |P_{G_2}| = p_k(G, v_1) + 1 = p_k(G_1, v_1) + p_k(G_2)$, $|P_{G_1}| = p_k(G_1, v_1)$ és $p_k(G_2) \leq p_k(G_2, v_2) \leq |P_{G_2}| = p_k(G_2)$, ahol az utolsó egyenlet első egyenlőtlensége a 4.2.1 állításból következik, a második egyenlőtlensége abból, hogy egy tetszőleges particionálás mérete mindig legalább akkora, mint az optimálisé, az utolsó egyenlőség pedig az előtte lévő egyenletből következik. Így tehát azt kapjuk, hogy $p_k(G_2) = p_k(G_2, v_2)$ és P_{G_i} optimális particionálás $i = 1, 2$ esetén is. Ezekből következik, hogy $l_k(G_1, v_1) \leq |q \cap V(G_1)| = 1$, tahát $l_k(G_1, v_1) = 1$, és $l_k(G_2, v_2) \leq |q \cap V(G_2)| < k$.

A másik irány bizonyításához tegyük fel, hogy $l_k(G_1, v_1) = 1$ és $l_k(G_2, v_2) < k$ és $p_k(G_2) = p_k(G_2, v_2)$. Legyen P_i egy optimális, gyökércsúccsal rendelkező k-út felosztása a G_i gráfnak és legyen bennül q_i azaz út, ami tartalmazza a v_i gyökércsúcsot $i = 1, 2$-re. Mivel $|q_1| = l_k(G_1, v_1) = 1$ és $|q_2| = l_k(G_2, v_2) \leq k$, ezért $(P_1 \cup P_2 \cup \{q_1 \cup \{v_1v_2\} \cup q_2\}) - \{q_1, q_2\}$ (ez igazából a $(P_1 \cup P_2 \cup \{\{v_1v_2\} \cup q_2\}) - \{q_2\}$-vel egyezik meg, hiszen a q_1 út csak a v_1 csúcsból áll, így az út felosztásban is a v_1 gyökércsúcs marad az új út végpontjaként) is egy k-útparticionálás lesz, melynek mérete $p_k(G_1, v_1) + p_k(G_2, v_2) - 1 = p_k(G_1, v_1) + p_k(G_2) - 1$, így $p_k(G, v_1) = p_k(G_1, v_1) + p_k(G_2) - 1$ szintén fentáll. □

4.2.5. Tétel. *Ha a G gráf felírható a G_1 és a G_2 gráf kompozíciójaként, amikben a v_1 és v_2 gyökércsúcsok, akkor a következőket mondhatjuk:*

$$l_k(G, v_1) = \begin{cases} l_k(G_2, v_2) + 1 & \text{ha } l_k(G_1, v_1) = 1 \text{ és } l_k(G_2, v_2) < k, \text{ és } p_k(G_2) = p_k(G_2, v_2) \\ p_k(G_1, v_1) + p_k(G_2) & \text{ha } p_k(G_2) \neq p_k(G_2, v_2) \\ \min\{l_k(G_1, v_1), l_k(G_2, v_2) + 1\} & \text{egyébként.} \end{cases} \tag{4.6}$$

Bizonyítás. A bizonyításhoz először is tegyük fel, hogy P egy optimális k-út particionálása G-nek és q azaz út benne, ami tartalmazza v_1 csúcsot, tehát a q meg fog egyezni az $l_k(G, v_1)$gyel.

Az első esetnél $l_k(G_1, v_1) = 1$ és $l_k(G_2, v_2) < k$, és $p_k(G_2) = p_k(G_2, v_2)$, az előző 4.2.4 tétel bizonyításából következik, hogy ekkor $l_k(G, v_1) = l_k(G_2, v_2) + 1$.

A következő esetben az $l_k(G, v_1) \leq l_k(G_1, v_1)$ egyenlőtlenség adódik a 4.2.2 állítás 4.3 egyenletéből. Ez után tegyük fel, hogy fentáll az, hogy $p_k(G_2) \neq p_k(G_2, v_2)$, pédául $p_k(G_2) = p_k(G_2, v_2) - 1$ a 4.2.2 állítás 4.2 egyenlete alapján.
Ha $q \cap V(G_2) \neq \emptyset$, akkor P_{G_i} egy gyökércsúccsal rendelkező k-út particionálása G_i-nek $i = 1, 2$ esetén. Ekkor $|P_{G_1}| + |P_{G_2}| = p_k(G, v_1) + 1 = p_k(G_1, v_1) + p_k(G_2) + 1 = p_k(G_1, v_1) + p_k(G_2, v_2)$. Ebből következik, hogy $|P_{G_1}| = p_k(G_1, v_1)$ és $l_k(G_1, v_1) = 1$. Tehát $l_k(G, v_1) = l_k(G_1, v_1) = 1$.
Ha $q \cap V(G_2) = \emptyset$, akkor $|P_{G_1}| + |P_{G_2}| = p_k(G, v) = p_k(G_1, v_1) + p_k(G_2)$. Ebből következik, hogy $|P_{G_1}| = p_k(G_1, v_1)$ és $l_k(G_1, v_1) \leq l_k(G, v_1)$, vagyis $l_k(G_1, v_1) = l_k(G, v_1)$

Az utolsó esetnél $l_k(G, v_1) \leq l_k(G_2, v_2) + 1$ mivel $p_k(G_2) = p_k(G_2, v_2)$.
Ha $l_k(G_1, v_1) = 1$, akkor $1 \leq l_k(G, v_1) \leq l_k(G_1, v_1) = 1$.
Tegyük fel, hogy $l_k(G_1, v_1) \neq 1$.Ekkor $|P_{G_1 - v_1}| = p_k(G_1, v_1)$ és $|P_{G_1}| = p_k(G_1, v_1)$.
Ha $q \cap V(G_2) = \emptyset$, akkor $l_k(G, v_1) = l_k(G_1, v_1)$.
Ha $q \cap V(G_2) \neq \emptyset$, akkor $l_k(G, v_1) = l_k(G_2, v_2) + 1$.
Tehát $l_k(G, v_1) = min\{l_k(G_1, v_1), l_k(G_2, v_2) + 1\}$. □

A fenti állítások és tételek valamint annak segítségével, hogy egy fa felépíthető két diszjunkt fa kompozíciójából felírhatjuk a következő algoritmust, ami megadja egy tetszőleges fa esetén az optimális k-út particióinak minimális számát.

k-út partíciók számának meghatározása fában

(0) Mélységi bejárással jelöljük meg a fa csúcsait $v_1, v_2, \dots, v_n$ címkékkel.

(1) Az algoritmus második lépésében a kezdeti értékek kerülnek beállításra:

- $p(v_i) := 1$ minden $i = 1 \dots n$ - ez jelöli $p_k(T_i)$-t, azaz a részfában a k-út particionálás minimális út számát,
- $p'(v_i) := 1$ minden $i = 1 \dots n$ - ez jelöli $p_k(T_i, v_i)$-t, azaz a részfában a v_i gyökércsúccsal rendelkező k-út particionálás minimális útszámát,
- $l(v_i) := 1$ minden $i = 1 \dots n$ - ez jelöli $l_k(T_i, v_i)$-t, azaz a v_i csúcsból induló út hosszát a részfában.

(2) Legyen a v_j csúcs őse a v_i csúcs. A következő lépéseket hajtsuk végre $i = 1 \dots n-1$-ig. Állítsuk be következő értéket $A := \{t : \ p'(v_t) = p(v_t),\ t = i \text{ vagy } j\ \}$

Értelmezés: A értékétől függ, hogy a 4.2.3 tétel szerint milyen érték határozható meg az optimumnak.

(2.1) Ha $|A| = 2$ és $l(v_i)+l(v_j) \leq k$, akkor $p(v_j) := p(v_j)+p(v_i)-1$, egyébként $p(v_j) := p(v_j)+p(v_i)$

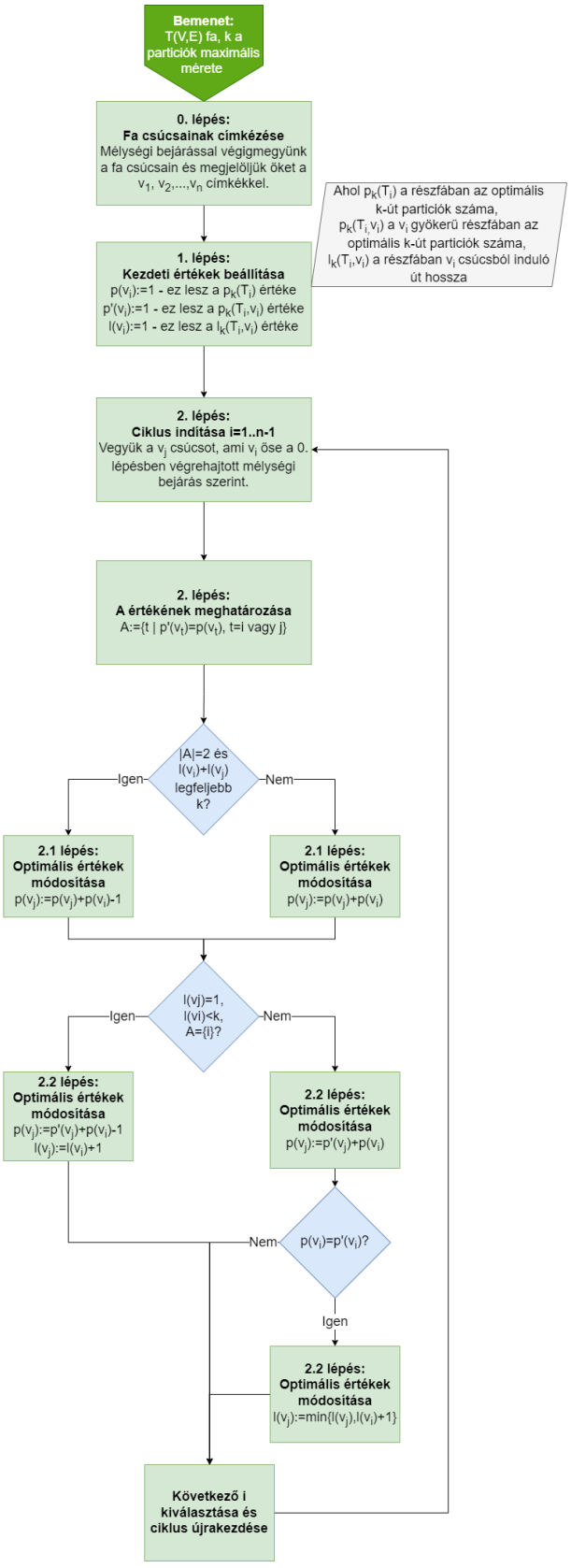
Értelmezés: a 4.2.3 tételben szereplő 4.4 egyenletből következnek az értékadások.

(2.2) Ha $l(v_j) = 1$ és $l(v_i) < k$ és $A = \{i\}$, akkor $p'(v_j) := p'(v_j) + p(v_i) - 1$, $l(v_j) := l(v_i) + 1$, egyébként $p'(v_j) := p'(v_j) + p(v_i)$ és ha $p(v_i) = p'(v_i)$, akkor legyen $l(v_j) := \min\{l(v_j), l(v_i) + 1\}$

Értelmezés $l(v_j)$ értékadásai a(z) 4.6 egyenletből következnek. A többi értékadás, pedig a(z) 4.5 egyenletből következik. Az algoritmus végén $p(v_n)$ megmondja, hogy mi a gráf minimális k-út particióinak a száma, azonban azt nem mondja meg, hogy hogyan lehet ennyi úttal lefedni a gráfot.

4.2.6. Tétel. *A fenti algoritmus lináris időben lefut és megmondja a T fa k-út particióinak minimális számát.*

Az algoritmus futását az alábbi folyamatábra is szemlélteti.



4.1. ábra. k-út partíciók számának meghatározása fában

4.3. k-út particionálás vizsgálata kaktuszokban

Szakdolgozatom következő részében kaktuszokat fogom vizsgálni. Ezen fejezet végén egy olyan algoritmus kerül bemutatásra, mely kaktuszokban megtalálja az optimális k-út particiót és a mérete mellett magát a particionálást is megadja. Ebben a fejezetben az irodalomjegyzékben található [6] számú cikk eredményit, tételeit, illetve az ott vázolt algoritmust fogom felhasználni.

Az irodalomjegyzék [3] számú cikkben egy speiális k-út particionálás kerül bevezetésre Steiner által, amit szakdolgozatom ezen részében Steiner-féle k-út particionálásnak fogok nevezni. (Ennek nincs köze a Steiner fákhoz.) A Steiner-féle k-út particionálás a következőképp néz ki: a $V(G)$ csúcshalmaz két diszjunkt részre D-re és S-re bontjuk fel, azaz $D \cup S = V(G)$ és $D \cap S = \emptyset$. A probléma az összes diszjunkt k-út megtalálása, ami feledi D-t, ez pontosan megegyezik a k-út particionálás problémájával ha $D = V(G)$. Az S halmazt Steiner-féle csúcshalmaznak nevezzük. Egy opimális Steiner-féle k-út particionálás tartalmazhat csúcsokat az S halmazból is a lefedetteken kívül. Jelölje $p_k(G, D)$ a minimális számát a csúcs diszjunkt k-utaknak, amik lefedik D-t, nevezzük ezt lefedési vagy particionálási számnak.

4.3.1. Definíció (Steiner k-út particionálás tartalmaz egy e élt). *Legyen S_G a G gráf Steiner k-út lefedése. Azt mondjuk, hogy S_G felhasznál egy $e \in E(G)$ csúcsot, ha S_G valamelyik útja tartalmazza e-t.*

4.3.2. Definíció (Függőút (pendant path)). *A $v_1v_2 \dots v_m$ utat v_1-ben kezdődőd függő útnak nevezzük, ha $m \geq 2$, $d(v_i) = 2$, $1 < i < m$ esetén és $d(v_m) = 1$.*

4.3.3. Definíció (Villa (fork)). *Legyen T egy gyökérrel rendelkező fa. Egy legalább két gyerekcsúccsal rendelkező T-beli csúcsot villának nevezünk, ha minden leszármazottjának a foka maximum 2. Egyértelmű, hogy minden gyökérrel meg nem egyező villának a foka legalább 3 a T-ben.*

A célunkat, a kaktusz k-út particionálását két művelet, eljárás segítségével fogjuk elérni. Ezek az eljárások az éltörlés és a kör nyitás.

Az első eljárás, amivel foglalkozunk az az éltörlés lesz. Ez az eljárás fákra is alkalmazható, ebben az esetben nem csak a partiók optimális számát kapjuk meg (ahogy azt a korábbi fejezetben láttuk), hanem magát a particionálást is. Az eljárás alapjait a következő állítások adják.

4.3.4. Állítás. *Legyen T egy gyökércsúccsal rendelkező fa. Ha van olyan x csúcs, melynek 1 a foka és $x \in S$, akkor $p_k(T, D) = p_k(T - x, D)$*

Bizonyítás. Legyen S_T egy optimális Steiner-féle k-út lefedése a gráfnak. Ha S_T nem fedi le x-et, akkor kész vagyunk. Ha S_T lefedi x-et, akkor tegyük fel, hogy $p \in S_T$ az az út, lefedi. Könnyen látható, hogy x csúcs az út végén van, így könnyen törölhető. Legyen $p* = p - x$. Ekkor ha S_T-ben lecseréljük p-t a $p*$ útra az továbbra is optimális Steiner k-út lefedése lesz T-nek és $T - x$-nek is. □

4.3.5. Állítás. *Legyen v_1 egy villa a gyökércsúccsal rendelkező T fában, ahol minden levélcsúcs eleme a D csúcshalmaznak. Ha $v_1v_2 \dots v_m$ egy v_1-ben kezdődő függőút, ahol $m > k$, akkor $p_k(T, D) = p_k(T - v_{m-k}v_{m-k+1}, D)$.*

Bizonyítás. A bizonyításhoz megkeressük azt a két k-utat. ami tartalmazza a függőút $v_{m-k}v_{m-k+1}$ élét és az utána lévő éleket majd átrendezzük az utakat úgy, hogy az egyiknek a v_{m-k}, a másiknak a v_{m-k+1} legyen a végén. Ezután, ha a v_{m-k}, v_{m-k+1} csúcsokat kitöröljük a gráfból nem változik meg a D halmaz lefedési száma. Egzakt leírással tegyük fel, hogy S_T optimális Steiner féle k-út lefedés tartalmazza a $v_{m-k}v_{m-k+1}$ élt. Legyen ez az él $p = p_1v_{m-k}v_{m-k+1}p_2$. Legyen $p_3 = v_{j+1}v_{j+2}\dots v_m$ a maradék csúcsokból álló k-út S_T-ben. Ezután rendezzük át az utakat a következő módon: legyen $p'_1 = p_1v_{m-k}$ és $p'_2 = v_{m-k+1}p_2p_3$. Ekkor az $S'_T = S_T - p - p_3 + p'_1 + p'_2$ továbbra is egy optimális Steiner féle k-út partició T-re és $T - v_{m-k}v_{m-k+1}$-ra egyaránt. □

4.3.6. Állítás. *Legyen v_1 egy villa a gyökércsúccsal rendelkező T fában, ahol minden levélcsúcs eleme a D csúcshalmaznak. Tegyük fel, hogy v_1-ből induló összes függőút maximum k csúcsot tartalmaz. Ha van két függőút $v_1v_2\dots v_m$ és $v_1^*v_2^*\dots v_n^*$, amik uniója maximum k csúcsot tartalamaz, akkor $p_k(T, D) = p_k(T - \{v_1x \in T(E) : x \neq v_2, x \neq v_2^*\}, D)$*

Bizonyítás. A bizonyításban először megkeressük azt az utat, ami jelenlegi felosztásban tartalmazza v_1-et, majd attól függően kettéágazik a bizonyítás, hogy ez az út tartalmazza-e v_2 vagy v_2^* valamelyikét.

1) Az első esetben feltesszük, hogy v_1v_2 élt tartalmazza az eredeti felosztás egy $p = p_1v_1v_2p_2$ útja, aminek a p_1 részébe esik a v_2 csúcs és a $p_3 = v_2^*\dots v_n^*$. Ezután a v_1 csúcson keresztül összefűzésre kerül a p_1 és p_3 utak és a p_2 út külön fog szerepelni a felosztásban. Látható, hogy $S'_T = S_T - p - p_3 + p_2 + p_1v_1p_3$ továbbra is optimális Steiner felosztása a T és $T - \{v_1x \in T : x \neq v_2, x \neq v_2^*\}$ gráfoknak egyaránt. (Ha $v_1v_2^*$-ot tartalmazná ugyanaz lenne a bizonyítás csak "tükrözve".)

2) Második esetben, azaz ha az eredeti S_T felosztás nem tartalmazza se a v_1v_2, se a $v_1v_2^*$ utakat, akkor mind v_2 és v_2* egy-egy különálló p_1 és p_2 út részei. Valamint van egy $p = p_3v_1p_4$ út, ami tartalmazza a v_1-et. Ekkor ha p_3 és p_4 külön utak lesznek és a p_1, p_2 út a v_1 csúcsnál összefűzésre kerül, akkor a kapott felosztás továbbra is optimális T-re és $T - \{v_1x \in T : x \neq v_2, x \neq v_2^*\}$ gráfoknak egyaránt. □

4.3.7. Állítás. *Legyen v_1 egy villa a gyökércsúccsal rendelkező T fában, ahol minden levélcsúcs eleme a D csúcshalmaznak. Tegyük fel, hogy több függőút is kiindul a v_1 csúcsból, amik legfeljebb k csúcsot tartalmaznak. Legyenek $u_1, u_2, \dots, u_m$ a v_1 csúcs gyerekei és legyen $v_1u_1\dots x$ a legrövidebb ezen függőutak közül. Ha bármely két függő út uniója több, mint k csúcsot tartalmaz, akkor $p_k(T, D) = p_k(T - \{v_1u_i : i = 2, 3, \dots, m\}, D)$.*

Bizonyítás. A bizonyítás itt is több esetből fog állni. Először tegyük fel, hogy S_T egy optimális Steiner-féle k-út felosztása T-nek. Az állítás egyik feltételéből adódóan egyértelmű, hogy nincs olyan k-út S_T-ben, ami egyszerre két függőutat fedne le. Tegyük fel, hogy a v_1u_i élt lefedi S_T valamilyen $i \geq 2$-re és legye p, ami ezt az élt fogja ($p = p_1v_1u_ip_2$). Tegyük fel, hogy $v_1u_ip_2$ egy v_1-ből induló függőút. Ezután két fő ágra bontható a bizonyítás:

1) Először nézzük azt az esetet, amikor p_1 nem tartalmazza a v_1 csúcs egyik leszármazottját sem. Ekkor a $p_3 = u_1\dots x$ egy útja lesz az S_T felosztásnak. Könnyen látható, hogy a p utat v_1u_i között elvágva és a v_1 csúcsban a p_3 úttal összekötve továbbra is optimális felosztást kapunk T-re és $T - \{v_1u_i : i = 2, 3, \dots, m\}$-re egyaránt. Tehát $S'_T = S_T - p - p3 + p_1v_1p_3 + u_ip_2$. A $p_1v_1p_3$ út biztosan nem hosszabb k-nál, hiszen a p_2 függőutat egy rövidebb vagy ugyanolyan hosszú úttal helyettesítettük.

2) Második esetben feltételezzük, $p_1 = y' \dots u_j$ úttal. Legyen $v_1 u_j \dots y'y" \dots y$ a v_1-ből induló függőút. Ekkor a $p_4 = y \dots y"$ egy út lesz S_T-ben. Ezen a ponton ismét két alesetre esik szét a bizonyítás:

2/a) Ha $j \neq 1$. Ebben az esetben is $p_3 = u_1 \dots x$ egy útja lesz az S_T felosztásnak. Ekkor a $p_1 p_4$ utakat összefűzve, p-t és p_3-at az egyes esethez hasonlóan átrendezve ismét egy optimális felosztást kapunk. A $p_1 p_4$ út egy v_1-ből induló függőút a v_1 csúcs nélkül, ezért az állítás feltételei szerint biztosan rövidebb k-nál. Tehát az új $S'_T = S_T - p_3 - p_4 + p_4 p_1 + v_1 p_3 + u_i p_2$ egy optimális Steiner-féle felosztása lesz T-nek és $T - \{v_1 u_i : i = 2, 3, \dots, m\}$-nek is.

2/b) A másik aleset, amikor $j = 1$, tehát az eredeti felosztásban is szerepelt már a $v_1 u_1$ él. Ekkor van egy $p_4 = x \dots y"$ út. Rendezzük át az utakat úgy, hogy az $v_1 u_1$ kezdetű függőél teljes egészében a felosztás egy útja és az eredeti p utat szokás szerint elvágjuk a $v_1 u_i$ élnél. Tehát a módosított felosztás a következő lesz $S'_T = S_T - p - p_4 + p_4 p_1 v_1 + u_i p_2$ optimális Steiner-féle felosztása lesz a T-nek és a $T - \{v_1 u_i : i = 2, 3, \dots, m\}$-nek. □

A fenti állítások alapján feltételezhetjük a következőket (mivel ha nem így van, akkor a fentiek szerint át tudjuk alakítani ilyenné):

•Minden levél csúcs eleme D-nek

•Minden függőút maximum k csúcsból áll T-ben.

•Tudnánk építeni egy Steiner-féle k-út particionálás fákra.

Vezessünk be egy új fogalmat, ahol a v csúcs kiemelt szerepet játszik:

4.3.8. Definíció (v-n függő fa (tree suspended at v)). *Legyen G egy kaktusz és legyen v csúcsot tartalmazza egy G-beli kör. Legyenek G_i-k $i \in I$ a $G - v$ gráf azon komponensei, melyre $G[V(G_i) \cup v]$ nem tartalmaz kört. A $T_v = \bigcup_{i \in I} G[V(G_i) \cup v]$ fát a v-n függő fának nevezzük.*
Jegyezzük meg, hogy ha az I indextömb üres, akkor T_v egyedül a v csúcsból áll.

A fákhoz hasonlóan a kaktuszokból képzett T_v fákra is kimondhatóak a korábbi tételeink. A tételek bizonyítása analóg módon megegyezik a korábban vázolt bizonyításokkal.

4.3.9. Állítás. *Legyen G egy kaktusz. Ha van olyan x csúcs, melynek 1 a foka és $x \in S$, akkor $p_k(T, D) = p_k(T - x, D)$.*

4.3.10. Állítás. *Legyen G egy kaktusz, ahol minden levélcsúcs eleme a D csúcshalmaznak. Legyen v_1 egy villa a T_v függő fában. Ha $v_1 v_2 \dots v_m$ egy v_1-ben kezdődő függőút, ahol $m > k$, akkor $p_k(T, D) = p_k(T - v_{m-k} v_{m-k+1}, D)$.*

4.3.11. Állítás. *Legyen G egy kaktusz, ahol minden levélcsúcs eleme a D csúcshalmaznak. Legyen v_1 egy villa a T_v függő fában. Tegyük fel, hogy v_1-ből induló összes függőút maximum k csúcsot tartalmaz. Ha van két függőút $v_1 v_2 \dots v_m$ és $v_1^* v_2^* \dots v_n^*$, amik uniója maximum k csúcsot tartalmaz, akkor $p_k(T, D) = p_k(T - \{v_1 x \in T(E) : x \neq v_2, x \neq v_2^*\}, D)$.*

4.3.12. Állítás. *Legyen G egy kaktusz, ahol minden levélcsúcs eleme a D csúcshalmaznak. Legyen v_1 egy villa a T_v függő fában. Tegyük fel, hogy több függőút is kiindul a v_1 csúcsból, amik legfeljebb k csúcsot tartalmaznak. Legyenek $u_1, u_2, \dots, u_m$ a v_1 csúcs gyerekei és legyen $v_1 u_1 \dots x$ a legrövidebb ezen függőutak közül. Ha bármely két függő út uniója több, mint k csúcsot tartalmaz, akkor $p_k(T, D) = p_k(T - \{v_1 u_i : i = 2, 3, \dots, m\}, D)$.*

A fenti állítások felhasználásával felírhatjuk az alábbi eljárást:

Eljárás: G **metszése**

(1) Ha van olyan $x \in G$ csúcs, aminek 1 a foka és $x \in S$, akkor töröljük az x csúcsot. A(z) 4.3.9 állításból következik, hogy ezen csúcsok törlése nem fogja befolyásolni a particionálást vagy annak méretét.

(2) Keressünk egy u csúcsot a gráfban, ahol u egy villa a jelenlegi kaktuszhoz tartozó valamely függő fában.

(2.1) Ha van olyan függőút, ami több, mint k csúcsból áll, akkor azt metsszük a 4.3.10 alapján, azaz kivesszük belőle a $v_{m-k}v_{m-k+1}$ élt. így az út maximum k csúcsból fog állni.

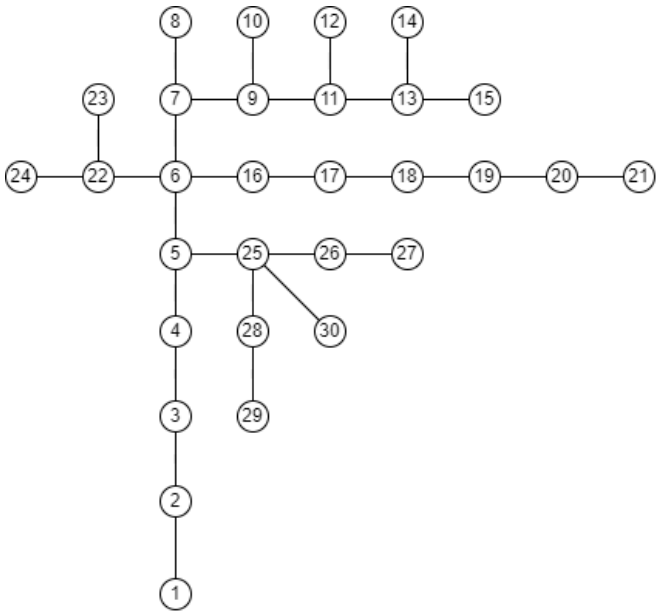
(2.2) Ha van két függőút, amik u-ban kezdődnek és az uniójuk maximum k csúcsból áll, akkor a gráfot metsszük a 4.3.11 alapján, azaz kitörölhetjük a u többi szomszédját, akik nem ehhez a két függőúthoz tartoznak és a particionálásunk továbbra is optimális marad.

(2.3) Ha van legalább két függő út, amik u-ból indulnak és bármely kettő uniója több, mint k csúcsból áll, akkor megmetsszük a 4.3.12 alapján, azaz a legrövidebb, u-ból induló függőutat megtartjuk, és u többi szomszédját, akikből függőút indul ki töröljük, A particionálás méretén nem változtat ezen csúcsok törlése.

Ezután ugorjunk vissza a (2) lépésre, ha még van villa a gráfunkban.

Ha ezt az eljárást egy fára alkalmazzuk, akkor megkapjuk az optimális k-út particionálását. Ezt egy konkrét példán be is mutatom. A példához alkottam egy random fát, amin az algoritmus futását fogom szemléltetni.

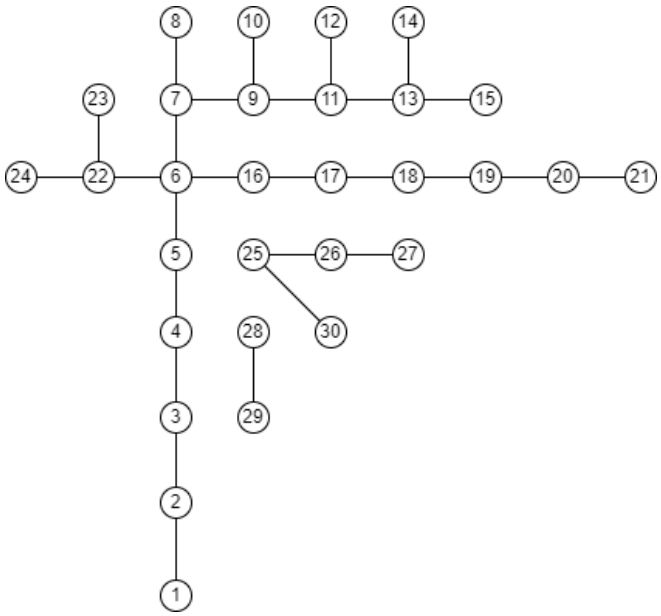
4.3.13. Példa. *Nézzük meg ezt a metszési eljárást egy konkrét fára és a* $k = 5$ *esetre. Legyen* $S = \emptyset$ *és nézzük a következő 4.2 ábrán felrajzolt fát:*



4.2. ábra. Fa

Mivel az S halmazt az üreshalmaznak választottuk meg annak érdekében, hogy az egész fát lefedjük a k-út particionálással, ezért az (1) lépést átugorjuk, nincs olyan csúcs, ami megfelelne az itt megfogalmazott feltételeknek.

A (2) lépésben legyen $u = v_{25}$ számú csúcs a villa. Ebből a csúcsból nem indul 5-nél hosszabb függőút, azért a (2.1)-es lépést nem hajtjuk végre. A (2.2)-ben megfogalmazott feltételeknek megfelel a 25-ös csúcs, indul belőle két olyan út, melyek unióka kisebb 5-nél, ezért ezt hajtjuk végre. Az út, amit megtartunk legyen a következő: $v_{27}v_{26}v_{25}v_{30}$ és a v_{25}-ből induló következő élek kerülnek törlésre: $v_{25}v_{30}, v_{25}v_5$. Ezután a fánk a következőképp fog kinézni: 4.3 ábra

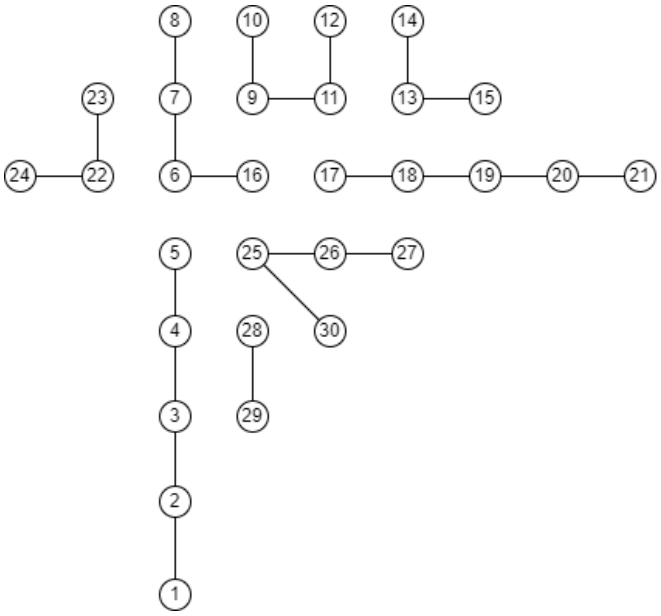


4.3. ábra. Első metszés után

Ugorjunk vissza a (2) lépésre és következő csúcsként $u = v_{22}$-t válasszuk. Ekkor ismét a (2.2) lépésben megfogalmazott feltételek fognak teljesülni és töröljük a $v_{22}v_6$ élt. Azután válasszuk u-nak a v_{13}-as csúcsot és a (2.2) lépés szerint töröljük a $v_{13}v_{11}$ élt. Majd legyen $u = v_9$ és a (2.2) lépés alapján töröljük a v_9v7 élét a fának.

Ezután a (2).es lépésben vegyük az $u = v_6$ csúcsot. Ekkor a (2.1) lépésben található feltételnek megfelelően van olyan függőút a csúcsból kiindulva, ami több, mint 5 csúcsból áll. Nézzük a $v_1v_2 \dots v_6$ utat. Ekkor az végétől 5 csúcsot leszámolva kell törölni a rákövetkező élt, tehát a v_5v_6 él kerül törlésre. Ha az $u = v_6$ marad továbbra is a (2.1) lépés feltételeinek felelünk meg, mert a $v_5v_{16}v_{17} \dots v_{20}$ út több, mint 5 csúcsból áll. Itt is az út végétől visszaszámolva a $v_{16}v_{17}$ élt törölhetjük.

Ezután a lépés után nem marad olyan villa a fában, ami ne két függőút, legfeljebb $k = 5$ hosszú uniója lenne, ezért megáll az algoritmus. A gráf felbontásra került 5-út particionálással, a particiók száma $p_5(G) = 8$. A végső particionálást az alábbi 4.4 ábra mutatja be.



4.4. ábra. 5-út particionálás

Ezzel bevezetésre kerül az éltörlés eljárása. A következőkben a környitás eljáráshoz tartozó állításokat vizsgáljuk, majd a két eljárást felhasználva kialakításra kerül az algoritmus, mely meghatározza az optimális k-út particionálást a kaktuszokban.

A környitási eljáráshoz ismét újabb definíciókat és állításokat vezetünk be.

4.3.14. Definíció (Metszett kaktusz (trimmed cactus)). *Egy G kaktuszt metszett kaktusznak hívunk, ha minden függő fája egy darab csúcsból vagy egy maximum k csúcsot tartalmazó függőútból áll.*

4.3.15. Definíció (Gráf utolsó köre (end cycle)). *Legyen G egy kaktusz és legyen benne C egy kör. Jelölje C_G az unióját a C körnek és az összes C csúcsain függő fáknak. C_G egy indukált részgráfja lesz G-nek és tartalmazni fog egy egyedi kört. A C kört a G gráf utolsó körének hívjuk, ha $C = G$ vagy ha C tartalmaz egy egyedi v csúcsot, ami szomszédos egy $V(G) \setminus V(C_G)$ csúccsal.*

4.3.16. Definíció (Horgony (anchor)). *Ha $C = G$-vel, akkor C minden v csúcsát horgonynak nevezzük. Egyébként aC azon egyedi v csúcsát nevezzük így, ami szomszédos egy $V(G) \setminus V(C_G)$ csúccsal.*

A következő eljárás a kaktusz metszésére szolgál, ahol G a gráf, C az utolsó köre G-nek, aminek e egy éle és v a horgony csúcsa. T_v pedig a v csúcson függő fa. Az eljárás nagyban megegyezik a korábban bemutatott metszési eljárással.

Eljárás: (C, e)

(1) Ha van olyan $x \in G$ csúcs, aminek 1 a foka és $x \in S$, akkor töröljük az x csúcsot. A(z) 4.3.9 állításból következik, hogy ezen csúcsok törlése nem fogja befolyásolni a particionálást vagy annak méretét.

(2) Legyen u egy villa a jelenlegi kaktuszhoz tartozó függő fában.

(2.1) Ha van olyan u-ból induló függőút, ami több, mint k csúcsból áll, akkor azt metsszük a 4.3.10 alapján azaz kivesszük belőle a $v_{m-k}v_{m-k+1}$ élt. így az út maximum k csúcsból fog állni.

(2.2) Ha $u \neq v$ és van két függőút, amik u-ben kezdődnek és az uniójuk maximum k csúcsból áll, akkor a gráfot a metsszük 4.3.11 alapján, azaz kitörölhetjük a u többi szomszédját, akik nem ehhez a két függőúthoz tartoznak és a particionálásunk továbbra is optimális marad.

(2.3) Ha van legalább két függő út, amik u-ból indulnak és bármely kettő uniója több, mint k csúcsból áll, akkor megmetsszük a 4.3.12 alapján, azaz a legrövidebb, u-ból induló függőutat megtartjuk, és u többi szomszédját, akikből függőút indul ki töröljük, A particionálás méretén nem változtat ezen csúcsok törlése.

(2.4) Ha pontosan 3 függő út indul ki v-ből a jelenlegi kaktuszban, és van két olyan út, aminek az uniója legfeljebb k csúcsot tartalmaz, akkor legyen $vx \dots y$ a leghosszabb függőút a három közül. Töröljük a vx élt. A(z) 4.3.11 állítás alapján ezesetben sem fog változni a particionálásunk optimális mérete.

A fenti eljárás minden lépésében elszeparáltunk néhány k-utat az aktuális kaktuszban.

Ezek után vezessük be az $S_e(C)$ jelölés, ami a fenti eljárással alkotott k-utakat jelöli. Az eljárás végén egy újabb kaktuszt kapunk, amit a továbbiakban G_e-vel jelölünk, amiben a T_v függő fa csak v csúcsból vagy egy maximum k hosszú függőútból áll vagy két függő út uniójából áll - melyek hossza összesen maximum k.

Mivel $C_G - e$ egy gyökércsúccsal rendelkező fa, ezért a korábbiakban vázolt állítások miatt kimondhatjuk a következőt:

$$p_k(C_G - e, D \cap V(C_G) = \begin{cases} |S_e(C)| & \text{ha } v \in S \text{ és a } T_v \in G_e \text{ egyedül a } v \text{ csúcsból áll} \\ |S_e(C)| + 1 & \text{egyébként.} \end{cases} \tag{4.7}$$

Ezután a T_v függő fa G_e kaktusz-beli elhelyezkedése alapján a C éleit a következő halmazokba sorolhatóak (minden e-re):

$$C_1 = \{e \in C : \text{a } T_v \text{ függő fa a } G_e \text{ kaktuszban egyedük a } v \text{ csúcsból áll}\}$$

$C_2 = \{e \in C : \text{a } T_v \text{ függő fa a } G_e \text{ kaktuszban egyetlen, maximum } k \text{ csúcsot tartalmazó függőútat tartalmaz}\}$

$C_3 = \{e \in C : \text{a } T_v \text{ függő fa a } G_e \text{ kaktuszban maximum két függőútat tartalmaz, amik uniója legfeljebb k csúcsból áll}\}$

Ekkor a három halmaz uniója kiadja a C kör összes élét és semelyik két halmaznak nincs közös éle, azaz metszetük az üres halmaz lesz.

Vezessük be az s_i értéket is a következő módon:

$$s_i = \begin{cases} \min_{e \in C_i}\{|S_e(C)|\} & \text{ha } C_i \neq \emptyset \\ \infty & \text{ha } C_i = \emptyset \end{cases} \tag{4.8}$$

Eztekből a következő eredményekre juthatunk:

$$p_k(C_G, D \cap V(C_G)) = \begin{cases} \min\{s_1 + 1, s_2 + 1, s_3 + 1\} & \text{ha } v \in D \\ \min\{s_1, s_2 + 1, s_3 + 1\} & \text{ha } v \in S \end{cases} \tag{4.9}$$

és

$$p_k(C_G - v, D \cap V(C_G - v)) = \min\{s_1, s_2 + 1, s_3 + 2\} \tag{4.10}$$

A következőkben feltételezhetjük, hogy $|S_{e_i}(C)| = s_i$, ahol $e_i \in C_i$ és $s_i \neq \infty$, és a v csúcsból induló függőút a G_{e_2} gráfban a lehető legkevesebb csúcsot tartalmazza, ha $s_2 \neq \infty$.

A következő állítások abban fognak segíteni, hogy eldönthessük, hogy a C kör mely élét törölve tudjuk úgy kinyitni a kört, hogy a lefedési számunkat is megőrizzük közben. Ehhez azonban újabb fogalmak és jelölések bevezetése szükséges.

Vegyük a v gyökércsúccsal rendelkező C_G részgráfot. Azt mondjuk, hogy a Steiner-féle k-út lefedés a v gyökérrel rendelkezik, ha van egy olyan út benne, aminek v az egyik vége. A v gyökérrel rendelkező Steiner-féle k-útlefedést a következőképp jelöljük: $p_k(C_G|v, D \cap V(C_G))$.

Jelölje az $l_k(C_G, v)$ a v-ből induló út minimális csúcsszámát, a v gyökérrel rendelkező Steiner-féle k-útfelosztásban.

4.3.17. Állítás. *Ha $min\{s_1 - 1, s_3\} \geq s_2$ akkor $S_{e_2}(C) + q$ egy optimális Steiner-féle k-útlefedése C_G részgráfnak úgy, hogy $|V(q)| = l_k(C_G, v)$, ahol q az egyedi, v-ből induló függőút G_{e_2}-ben, akkor a következő egyenőség teljesül:*

$$p_k(C_G|v, D \cap V(C_G)) = \begin{cases} s_3 + 2 & \text{ha } \min\{s_1 - 1, s_2\} > s_3\ , \\ s_2 + 1 & \text{ha } \min\{s_1 - 1, s_3\} \geq s_2, \\ s_1 + 1 & \text{egyébként.} \end{cases} \tag{4.11}$$

4.3.18. Állítás. *Ha $\min\{s_1 - 1, s_2\} > s_3$, akkor $p_k(G, D) = p_k(G - e_3, D)$.*

4.3.19. Állítás. *Ha $\min\{s_2, s_3\} > s_1 - 1$, akkor $p_k(G, D) = p_k(G - e_1, D)$.*

4.3.20. Állítás. *Ha $\min\{s_1 - 1, s_3\} \geq s_2$, akkor $p_k(G, D) = p_k(G - e_2, D)$.*

4.3.21. Állítás. *Ha $s_1 - 1 = s_3 < s_2$, akkor $p_k(G, D) = p_k(G - e_3, D)$ vagy $p_k(G, D) = p_k(G - e_1, D)$.*

4.3.22. Állítás. *$p_k(G, D) = p_k(G', D') + s_1$, ahol $G' = G_{e_1}$ és $D' = (D - v) \cap V(G')$.*

Egy optimális Steiner-féle k-út particionálás a kaktuszban

(1) Kezdeti értékek beállítása: $S_G = \emptyset$, $L = \emptyset$, $S_e(C) = \emptyset$ minde C kör e élére.

(2) Végezzük el a G metszése eljárást az aktuális G gráfunkon és

$$L := L \cup \{x : \text{x meg lett jelölve és törölve lett a } G \text{ metszése eljárás (1) lépése során}\}$$

(3) Ha G minde komponense egy k-út, akkor ugorjunk a (9) lépésre.

(4) Legyen C egy utolsó köre G-nek, amiben v a horgony csúcs.

(4.1) A C kör minden e élére végezzük el az Eljárás (C, e)-t és frissítsük az $S_e(C)$ értékeit minden e esetén megfelelően.

(4.2) Határozzuk meg az s_i-k értékét és az értékekhez tartozó e_i éleket $i = 1, 2, 3$-ra a C körben.

(5) Ha $\min\{s_1 - 1, s_2\} > s_3$, akkor $G := G - e_3$ és térjünk vissza a (2) lépéshez.

(6) Ha $\min\{s_2, s_3\} > s_1 - 1$, akkor $G := G - e_1$ és térjünk vissza a (2) lépéshez.

(7) Ha $\min\{s_1 - 1, s_3\} \geq s_2$, akkor $G := G - e_2$ és térjünk vissza a (2) lépéshez.

(8) Ha $s_1 - 1 = s_3 < s_2$, akkor jelöljük meg a v csúcsot az $l(C)$ jelöléssel, $G := G_{e_1}$, $D := (D - c) \cap V(G_{e_1})$ és $S := (V(G_{e_1}) \cap S) \cup \{v\}$ és térjünk vissza a (2) lépéshez.

(9) Minden $x \in L$ elemre, ami meg van jelölve az $l(C)$ jelöléssel $G := D \cup S_{e_3}(C)$.
Minden $y \notin L$, ami meg van jelölve az $l(C)$ jelöléssel $G := \cup S_{e_1}(C)$.

4.3.23. Tétel. *A fenti algoritmus egy optimális Steiner-féle k-út lefedését adja a G gráfnak polinomiális időben és a komplexitásának felső korlátja $O(n^2)$.*

4.4. k-particionálás vizsgálata páros permutációs gráfok esetén

Szakdolgozatom ezen szakaszában az irodalomjegyzék [2] számú cikkében bemutatott tételek és algoritmus segítségével a páros permutációs gráfokkal és a bennük való k-út particionálás algoritmusával fogok foglalkozni, azonban ehhez először különböző fogalmakat és állításokat vezetek be és vizsgálok meg. Ezen tételek fogják bizonyítani azt, hogy a fejezet végén található algoritmus valóban egy optimális k-út particionálását adja a páros permutációs gráfoknak.

A páros permutációs gráfokat az összehasonlítási gráfok egy részhalmaza. Ezen G=(X,Y;E) gráfoknál a csúcsok két osztályán erős rendezés állapítható meg $X = \{x_1, x_2, \ldots, x_r\}$ és $Y = \{y_1, y_2, \ldots, y_s\}$, Spinrad bizonyította, hogy ezek a rendezések liáris időben megtalálhatóak, ezért a továbbiakban feltételezzük, hogy ez az erős rendezés adott a gráf csúcsainak X és Y osztályára is. A páros permutációs gráfok a következő tulajdonsággal bírnak: ha $x_iy_l, x_jy_m \in E$ és $i < j, l > m$, akkor $x_iy_m, x_jy_l \in E$ is teljesül. Ebből a következő definíciók, elnevezések jönnek:

4.4.1. Definíció (Keresztező él). *Keresztező éleknek nevezzük azon $x_iy_l, x_jy_m \in E$ éleket, ahol $i < j$ és $l > m$.*

4.4.2. Definíció (Egymást keresztező utak). *Két út P_1 és P_2 keresztezi egymást, ha tartalmaznak keresztező éleket.*

Az erős rendezettségből adódóan értelmezhető a balról jobbra rendezés, haladás a páros permutációs gráfokban és természetesen adódnak a legbaloldalibb és legjobboldalibb kifejezések a csúcsokra, illetve az utakra is.

Az definíciók bevezetése után az algoritmus alapjául szolgáló állításokkal fogok foglalkozni.

4.4.3. Állítás. *Ha P_1, P_2 két út, melyek keresztező élt tartalmaznak G=(X,Y;E) páros permutációs gráf k-útparticiójában, akkor létezik két olyan csúcs-diszjunkt, keresztezésmentes P_1', P_2' út, ami ugyanazon csúcsokat fedi le, mint P_1 és P_2 és a hosszuk maximum k.*

Bizonyítás. A bizonyításban a gráf erős rendezetségéből indulunk ki. Az erős rendezettségből következően a P_1 és P_2 utak csúcsai is erősen rendezhetőek. Ezután feltesszük, hogy a_1b_1, a_2b_2 a legbaloldalibb és

c_1d_1, c_2d_2 a legjobboldalibb élek, amik keresztezik egymást és P_1 és P_2 élei. Az erős rendezésből következően van a gráfban megy út a_1, a_2 között (a_1, b_1, a_2) és d_1, d_2 között is (d_1, c_1, c_2). Ezek segítségével látható, hogy létezik egy kör, ami $C = a_2 -> a_1, P_1(bizonyoslei), d_1 -> d_2\overleftarrow{P_2}(bizonyoslei)a_2$ felhasználásával.

Ez a C kör egy Hamilton kört ad ki a $G[X_C \cup Y_C]$ csúcsok által indukált részgráfban, ahol $X_C \cup Y_C$ a C kör által feszített részgráf. Ezután Brandstät korábbi eredményeit használjuk fel, miszerint a páros permutációs gráfban akkor és csak akkor lehet Hamilton út, ha $x_iy_ix_{i+1}y_{i+1}$ egy négy hosszú kört adnak ki minden $1 \leq i < |X| = |Y|$ esetén.

Ez alapján a gráf erőst rendezést kihasználva $X_C \cup Y_C$-re, és a_1b_1, a_2b_2 meghatározásából adódóan, azaz hogy ők a legbaloldalalabbi keresztélek a P_1, P_2 utakban, a P_1 és P_2 közül csak az egyiknek lehet éle ezen élektől balra. Ezt az élt, ami P_1-ben vagy P_2-ben az a_1b_1, a_2b_2 utaktól balra helyezkedik el P_r-rel jelöljük. Ugyanígy a jobb oldalra is P_1 és P_2 közül csak az egyiknek lehet éle a c_1d_1, c_2d_2 élektől jobbra, melyet P_s-sel jelöl.

Tegyük fel, hogy P_r az x_{i_1} pontban lép be és P_s az y_{i_j} pontban lép ki az $X_C \cup Y_C$ részgráfból, ahol $X_C = \{x_{i_1} \dots x_{i_j}\}$, $Y_C = \{y_{i_1} \dots y_{i_j}\}$ az indukált részgráf erősen rendezett csúcsai. Ezek után már megadható a két új út, melyek ki fogják elégíteni az állításunkat. A két új út a következőképp fog kinézni:

(a) $P_1' = P_r x_{i_1} y_{i_1} x_{i_2} \dots x_{i_{\lceil (k-p)/2 \rceil}}$ és $P_2' = \overleftarrow{P_s} y_{i_j} x_{i_j} y_{i_{j-1}} \dots y_{i_{\lceil (k-p)/2 \rceil}}$, ha $k-p$ páratlan,

(b) $P_1' = P_r x_{i_1} y_{i_1} x_{i_2} \dots x_{i_{(k-p/2)}} y_{i_{(k-p/2)}}$ $P_2' = \overleftarrow{P_s} y_{i_j} x_{i_j} y_{i_{j-1}} \dots x_{i_{((k-p)/2)+1}}$, ha $k-p$ páros. □

Az állítást többször alkalmazása esetén az összes keresztél eltávolítható, helyettesíthető keresztező él mentes utakkal, ezért kimondhatjuk a következőt:

4.4.4. Következmény. *Legyen $\mathcal{P} = \{P_1, P_2, \dots P_t\}$ a G=(X,Y;E) gráf egy minimális csúcs diszjunkt k-út particiója, ekkor létszik egy keresztező él mentes $\mathcal{P}' = \{P_1', P_2' \dots, P_t'\}$ k-útptarciója is G-nek.*

Következő állítás előtt újabb definíció bevezetésére van szükségünk.

4.4.5. Definíció (Folyamatos út (contiguous path)). *Egy utat folyamatosnak nevezünk, ha az előáll $u_iv_ju_{i+1}v_{j+1}$.... alakban, ahol u-k és v-k X és Y erős rendezésében egymást követő csúcsok.*

A folyamatosságot felhasználva a következő állítást mondhatjuk ki:

4.4.6. Állítás. *Legyen $\mathcal{P} = \{P_1, P_2, \dots, P_t\}$ egy minimális, keresztél mentes csúcs-diszjunkt k-út paticiója G páros permutációs gráfnak. Ekkor létezik egy olyan $\mathcal{P}' = \{P_1', P_2', \dots, P_t'\}$ csúcs-diszjunkt k-út particiója is G-nek, amiben az utak folyamatosak.*

Bizonyítás. Mivel feltettük, hogy az eredeti $\mathcal{P}$ k-út felosztás keresztélmentes, ezért ha van nem folyamatos út, akkor az átugrik egy csúcsot, ami önálóan alkot egy particiót. Ezután az "ugrásnál" kicseréjük a két csúcsot úgy, hogy az átugrott csúcsot bevesszük abba a particióba, amiben a folytonosság miatt kell lennie, és a másik csúcsot (ami miatt nem volt folyamatos az eredeti particó) a másik particióba tesszük. Ezeket a lépéseket ismételve eljutunk az állításban kimondott állapotig, azaz, hogy minden felosztás folyamatos lesz.

A bizonyítás egzakt kifejezésekkel a következőképp írható le. Jelölje $P_l \in \mathcal{P}$ azt az utat, ami nem folyamatos. Tehát ebben az útban kell lennie egy olyan $u_iv_ju_f$ résznek, ahol $f > i+1$. Mivel a $\mathcal{P}$ k-út

felosztás a gráf összes csúcsát lefedi, ezért léteznie kell egy olyan P_g particíonak, ami csak az u_{i+1} csúcsból áll. G összefüggő gráf, ezért adódik, hogy az u_{i+1} csúcsnak biztosan van legalább egy v szomszédja, ahol a v csúcs a gráf másik csúcshalmazának a része (azaz ha $u_{i+1} \in X$, akkor $v \in Y$ vagy fordítva). Ekkor az $u_{i+1}v$ él vagy keresztezi az $\{u_iv_j, v_ju_f\}$ élek valamelyikét vagy $v_j = v$. Akármelyik esetről is van szó biztos, hogy $u_{i+1}v_j$ is egy éle lesz a gráfnak. Így a $\mathcal{P}$ felosztásban lévő P_l és P_g utakat kicserélhetjük a következő utakra: $P'_l = P_lv_ju_{i+1}$ és $P'_G = u_fP_l$. $\square$

A korábbiakból következik, hogy ha adottak csúcs diszjunkt folyamatos utak egy halmaza, akkor azok balról jobbra sorbarendezhetők, ahogy az őket alkotó csúcsok is. Így a halmaz utolsó eleme is jól definiálható, mint az utolsó út utolsó eleme.

Újabb jelölések és elnevezések vezetek be, melyek az algoritmus leírása során fogok felhasználni.

- Jelölje $N(x_i, y_l)$ a minimális, folyamatos k-út felosztását a generált $G[\{x_1, x_2, \ldots, x_i\} \cup \{y_1, y_2, \ldots, y_l\}]$ gráfban, ahol az utolsó fedett csúcs y_l, minden $1 \leq i \leq r$ és $1 \leq l \leq s$ esetén.
- Hasonlóan legyen $N(y_l, x_i)$ a minimális, folyamatos k-út felosztását a generált $G[\{x_1, x_2, \ldots, x_i\} \cup \{y_1, y_2, \ldots, y_l\}]$ gráfban, ahol az utolsó fedett csúcs x_i, minden $1 \leq i \leq r$ és $1 \leq l \leq s$ esetén.
- Egy v csúcsból induló folyamatos utat maximálisnak nevezünk, ha ez a v-ből induló folyamatos utak közül a leghosszabb, aminek a hossza nem nagyobb k-nál. Jelöljük ezt az utat $P(v)$-vel, hosszát pedig $|P(v)|$-vel.
- Jelölje $lastP(v)$ a $P(v)$ út utolsó csúcsát, $(lastP(v))^-$ az előtte lévő csúcsot és $(lastP(v))^{-2}$ az azt megelőzőt.

A korábbiak segítségével megalkothatunk egy olyan algoritmust, amiben a particiók folyamatos, kereszteződésmentes utakat fognak tartalmazni.

Steiner [2] algoritmusa minimális k-út particionálásra páros permutációs gráfban

(1) Első lépésben bevezetésre kerül két mesterséges csúcsot x_0, y_0, amik az X és Y csúcshalmazokhoz kerülnek hozzáadásra úgy, hogy a halmazok lebaloldalibb csúcsai lesznek és $x_0y_1, y_0x_1 \in E$. Ezután beállítjuk a változók, halmazok kezdeti értékeit. Az Elért csúcspárok $=\{(x_0, y_0), (y_0, x_0)\}$, $N(x_0, y_0) = 0$, $N(y_0, x_0) = 0$, $\mathcal{P}(x_0, y_0) = \emptyset$, $\mathcal{P}(y_0, x_0) = \emptyset$. Legyen $N(u_i, v_j) = \infty$, minden $(u_i, v_j) \neq (x_0, y_0)$ csúcspárra (ahol (u_i, v_j) az általános csúcspárt jeleöli $u_i \in X$ és $v_j \in Y$ vagy $u_i \in Y$ és $v_j \in X$).

(2) Második lépésben megkeressük és kivesszük az elért csúcspárok közül a legbaloldalabbi (u_i, v_j) csúcspárt és azt vizsgáljuk.

Ha $(u_i, v_j) = (x_s, y_r)$ vagy (y_s, x_r), akkor a (4)-es lépésre ugrunk, az algoritmus véget ér, mert elértük G végét.

Ha $u_i = x_r$ vagy y_s, akkor a (3)-as lépésre ugrunk, mert az egyik osztályban elértük az utolsó csúcsot, de még lehetnek fedetlen csúcsok a gráfban.

Ezután u_{i+1}-ből indulva megkeressük a leghosszabb utat, azaz megkeressük $P(u_{i+1})$-t és ezt vizsgáljuk.

(2.1) Ha $P(u_{i+1})$ út egy csúcsból, azaz csak u_{i+1}-ből áll és $N(v_j, u_{i+1}) > N(u_i, v_j) + 1$, akkor $N(v_j, u_{i+1}) = N(u_i, v_j) + 1$, $\mathcal{P}(v_j, u_{i+1}) = \mathcal{P}(u_i, v_j) \cup P(u_{i+1})$, és adjuk hozzá a (v_j, u_{i+1}) csúcspárt az elért csúcspárok halmazába. Ismételjük a (2)-es lépést.

Értelmezés: ha u_{i+1}-ből indulva csak az önmagából álló 1 hosszú út van gráfban, akkor megvizsgáljuk, hogy a v_j, u_{i+1} csúcsokig generált gráfban a minimális útszám (folyamatos, k-út particionálásban)

nagyobb-e, mint a megelező csúcsig lévő útszám +1 (azaz, u_i, v_j által generált és még egy út, ami az izolált u_{i+1} lenne). Ha nagyobb, akkor ezt adjuk neki értékül ($N(v_j, u_{i+1}) = N(u_i, v_j) + 1$) és a particiónk pedig úgy fog kinézni, hogy az előtte lévő csúcsig meghatározott partícióhoz hozzáadjuk még az u_{i+1}-ből induló leghosszabb utat (azaz az izolált u_{i+1} csúcsot).

(2.2) Ha $|P(u_{i+1})| > 1$, és $N(lastP(u_{i+1})^-, lastP(u_{i+1})) > N(u_i, v_j) + 1$, akkor legyen
$N(lastP(u_{i+1})^-, lastP(u_{i+1})) = N(u_i, v_j) + 1$
és $\mathcal{P}(lastP(u_{i+1})^-, lastP(u_{i+1})) = \mathcal{P}(u_i, v_j) \cup P(u_i + 1)$
és tegyük $(lastP(u_{i+1})^-, lastP(u_{i+1}))$ csúcspárt az elért csúcspárok halazába.
Értelmezés: Ha az u_{i+1}-ből induló leghosszabb út hosszabb, mint 1 csúcs, akkor megvizsgálásra kerül, hogy az út utolsó két csúcsában végződő utak minimális száma nagyobb-e, mint az eddigi minimális útszám+1. Ha nagyobb, akkor az eddigi minilális útszám+1 lesz az új értéke és az új partició, ami a megtalált út utolsó két csúcsáig generált gráfban van megegyezik a korábbi partícióval, amihez hozzáadjuk az új utat, ami u_{i+1}-ből indul.

(2.3) Ha $|P(u_{i+1})| = k$, és $N(lastP(u_{i+1})^{-2}, lastP(u_{i+1})^-) > N(u_i, v_j) + 1$, akkor legyen
$N(lastP(u_{i+1})^{-2}, lastP(u_{i+1})^-) = N(u_i, v_j) + 1$
és $\mathcal{P}(lastP(u_{i+1})^{-2}, lastP(u_{i+1})^-) = \mathcal{P}(u_i, v_j) \cup (P(u_{i+1}) \setminus u_{i+1})$,
a $(lastP(u_{i+1})^{-2}, lastP(u_{i+1})^-)$ csúcspárt tegyük az elért csúcspárok halmazába.
Értelmezés: Ha az u_{i+1}-ből induló maximális út hossza k, azaz a lehető leghosszabb, akkor a korábbi lépésben látott vizsgálatokat az út utolsó előtti és az előtti csúcsára végezzük el és a parcióhoz is a talált utat az utolsó (u_{i+1}) csúcsot levágva tesszük be. Erre azért van szükség, mert így u_{i+1} lehet az összekötő a másik oldali, még le nem fedett csúcshoz.

(3) Ebben a lépésben már feltételezzük, hogy már csak l db csúcs van az egyik oldalon, ami még nincs lefedve a particionálással. Minden csúcsot egyesével hozzáadunk a $\mathcal{P}(u_i, v_j)$ particóhoz, mint egy hosszúságú utak. Így a végső particiónk $\mathcal{P}(x_r, y_s)$ lesz, amiben $N(x_r, y_s) = N(u_i, v_j) + l$ darab út van, ha $u_i = x_r$ és $N(x_r, y_s) > N(u_i, v_j) + l$ vagy a parcitió $\mathcal{P}(y_s, x_r)$ lesz, amiben $N(y_s, x_r) = N(u_i, v_j) + l$ darab út van, ha $u_i = y_s$ és $N(y_s, x_r) > N(u_i, v_j) + l$. Adjuk hozzá a megfelelő (x_r, y_s) vagy (y_s, x_r) csúcspárt az elért csúcspárok halmazához és ismételjük meg a (2)-es lépést.

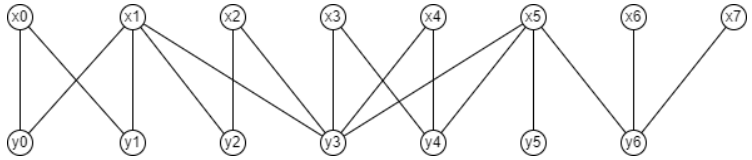
(4) Az algoritmus a végéhez ért, G minden csúcsát elértük. Legyen $p_k(G) = min\{N(x_r, y_s), N(y_s, x_r)\}$. Az algoritmus kimenete a $p_k(G)$ szám és az utak, amin ezt a minimumot elértük.

4.4.7. Tétel. *Steiner [2] algoritmusa megtalálja $p_k(G)$-t és a minimális k-út particionálást a $G = (X, Y; E)$ páros permutációs gráfban $O(n^2)$ időben és térben, ahol $n = |X| + |Y|$.*

Bizonyítás. Az algoritmus helyessége a korábbi tételekből, állításokból következik. Az algoritmus dinamikus programozással G-n végigmenve ki tudja számolni $N(u_i, v_j)$-t balról jobbra haladva az erős rendezésen. Mivel maximum $|X| * |Y| \leq n^2$ csúcspár van és minden lépés $O(1)$ idő alatt lefut egy csúcspárra, így a megkövetelt futási idő is előáll. □

Az algoritmus futását egy konkrét példán keresztül is szeretném bemutatni. Ehhez alkottam egy páros permutációs gráfot, amin végig fogom vinni az algoritmus egyes lépéseit.

4.4.8. Példa. *A pélában vegyük a következő 4.5 ábrán látható gráfot és vállaszuk meg $k = 5$-öt, mint maximális partició méret.*



4.5. ábra. Páros permutációs gráf (x_0, y_0) mesterséges csúcsokkal kiegészítve

Az algoritmus első lépése az (x_0, y_0) mesterséges csúcsok felvétele, mely az ábrán is már berajzolásra került. Valamint vegyük fel a kezdeti érékeket is az algoritmus első lépésének megfelelően.

Ezután hajtsuk végre a (2) lépést, azaz vegyük az elért csúcspárok legbaloldalabbit, ami az (y_0, x_0) csúcspár. Ezt vegyük ki a halmazból és vizsgáljuk meg. Mivel egyik csúcsa se egyezik meg a gráf legjobboldalabbi csúcsaival, ezért keressük meg az y_1 csúcsból induló leghosszabb, legfeljebb 5 hosszúságú folyamatos utat: $P(y_1) = y_1x_1y_2x_2y_3$. Mivel ez az út nem egy hosszú, ezért a (2.1) lépést átugorjuk és a (2.2) lépést nézzük. Az első feltétel, hogy $|P(y_1)| > 1$ teljesül, a második feltételünk $\infty = N(x_2, y_3) > N(y_0, x_0) + 1 = 1$ szintén telesül, ezért elvégezzük az értékadásokat. $N(x_2, y_3) := 1$, $\mathcal{P}(x_2, y_3) := \{y_1, x_1y_2x_2y_3\}$ és a (x_2, y_3) csúcspár kerüljön be az elért csúcspárok halmazába. Ezután a (2.3) lépésre ugrunk, ahol szintén mind a két feltétel teljesül, vagyis $P|(y_1)| = 5$ és $\infty = N(y_2, x_2) > N(y_0, x_0) + 1 = 1$, ezért az ebben a lépésben írtakat is végrehajtjuk. $N(y_2, x_2) := 1$, $\mathcal{P}(y_2, x_2) := \{y_1, x_1y_2x_2\}$. Ezután visszaugrunk a (2) lépésre.

A legbaloldalabbi elért csúcspárunk az (x_0, y_0) lesz, ezt kivesszük a halmazból és megvizsgáljuk. Indítunk egy folyamatos utat x_1 csúcsból, $P(x_1) = x_1y_2x_2y_3x_3$. Mivel ez nem csak egy csúcsból áll, így ugorjunk a (2.2) lépésre. $\infty = N(y_3, x_3) > N(x_0, y_0) + 1 = 1$, ezért végrehajtjuk az értékadásokat. $N(y_3, x_3) := 1$, $\mathcal{P}(y_3, x_3) := \{x_1y_2x_2y_3x_3\}$ és (y_3, x_3) legyen az elért csúcspárok között. Majd a (2.3) lépés feltételeit vizsgáljuk $|P(x_1)| = 5$ teljesül, azonban $1 = N(x_2, y_3) > N(x_0, y_0) + 1 = 1$ nem teljesül, ezért nem hajtjuk végre ebben a lépésen az értékadásokat, hanem ismét a (2) lépésre ugrunk.

Kivesszük az elért csúcspárok halmazából a legbaloldalabbit, azaz az (y_2, x_2) csúcspárt. Mivel ez még nem a gráf vége ismét keresünk egy folyamatos utat $P(y_3) = y_3x_3y_4x_4$. Az út hossza miatt most biztosan csak (2.2) lépést kell vizsgálnunk. $\infty = N(y_4, x_4) > N(y_2, x_2) = 1$, ezért az értékadást is végrehajtjuk. $N(y_4, x_4) := 2$, $\mathcal{P}(y_4, x_4) := \{y_1, x_1y_2x_2, y_3x_3y_4x_4\}$ és (y_4, x_4) kerüljön az elért csúcspárok közé, majd ismét a (2) lépés következik.

Következő vizsgálandó csúcspár az (x_2, y_3). Az x_3-ból indított utunk a $P(x_3) = x_3y_4x_4$ lesz. Mivel ez egy 3 hosszú út, ezért csak a (2.2) lépés másik feltételét kell vizsgálnunk. Mivel $2 = N(y_4, x_4) > N(x_2, y_2) + 1 = 2$ nem igaz, ezért a további lépéseket nem hajtjuk végre, hanem a (2) lépéssel folytatjuk.

Legbaloldalabbi csúcspárunk az (y_3, x_3), ami nem a gráf vége, ezért utat indítunk a rákövetkező csúcsból. $P(y_4) = y_4x_4$, ami szintén a (2.2) lépés első feltételének felel csak meg. A második feltétel ebben az esetben sem fog teljesülni, mert $2 = N(y_4, x_4) > N(y_3, x_3) + 1 = 2$ nem áll fent, ezért megkeressük a következő elért csúcspárt.

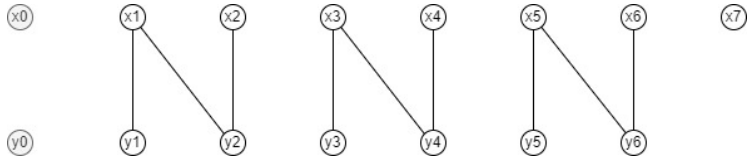
Ez a csúcspár az (y,x_4) lesz, ezért ismét egy folyamatos utat keresünk. $P(y_5) = y_5x_5y_6x_6$ egy négy hisszú út, úgyhogy a (2.1) és (2.3) lépéseket biztosan nem hajtjuk végre, a (2.2) lépés feltételét megvizsgáljuk. $\infty = N(y_6, x_6) > N(y_4, x_4) + 1 = 3$, ezért az értékadásokat elvégezzük: $N(y_6, x_6) := 3$,

$\mathcal{P}(y_6, x_6) := \{y_1, x_1y_2x_2, y_3x_3y_4x_4, y_5x_5y_6x_6\}$ *és az* (y_6, x_6) *csúcspárt hozzáadjuk az elért csúcspárok halmazához.*

Következő, legbaloldalabbi vizsgálandó csúcspár az (y_6, x_6)*, aminek első tagja a gráf legvége a csúcs osztályában, viszont a második tag nem egyezik meg a csúcsosztály utolsó tagjával, ezért a (3) lépést hajtjuk végre, azaz megkeressük az* X *osztályában a szabadon lévő, izolált csúcsokat. Csak egy ilyen csúcsot a találunk az* x_7 *csúcsot. Ezt izolált csúcsként hozzáadjuk a* $\mathcal{P}(y_6, x_6)$ *halmazunkhoz, így a következő párosítást kapjuk:* $\mathcal{P}(y_6, x_7) := \{y_1, x_1y_2x_2, y_3x_3y_4x_4, y_5x_5y_6x_6, x_7\}$*, illetve* $N(y_6, x_7) := N(y_6, x_6)+1 = 4$ *értéket is felülírjuk. Ezután ismét a (2) lépést hajtjuk végre.*

Az elért csúcspárok halmazában már csak az (y_6, x_7) *szerepel, ami a gráfunk utolsó két csúcsa, ezért az algoritmus végére értünk, a (4) lépést hajtjuk végre, ami alapján* $p_4(G) = \min\{N(x_7, y_6), N(y_6, x_7)\} = \min\{\infty, 4\} = 4$*, azaz* 4 *particióra esik szét a gráfunk az* 5*-út particionálás végén, amik a következő particiókból állnak:* $\mathcal{P}(y_6, x_7) := \{y_1, x_1y_2x_2, y_3x_3y_4x_4, y_5x_5y_6x_6, x_7\}$.

A kapott particiókat az 4.6 ábra is szemlélteti.



4.6. ábra. 5-út particionálás páros permutációs gráfban

5. fejezet

Konklúzió

A kiinduló üzleti problémánk az utasok csoportokra való bontása, particionálása volt, hogy elférjenek egy autóban. Ezt a problémát a k-út particionáláson keresztül közelítettem meg, mivel ha a particionálást élsúlyozott gráfban hajtjuk végre úgy, hogy a legkisebb költségű particionálást keressük, akkor ez megoldást adna az utasok szétosztására is úgy, hogy az összköltség minimális legyen. Szakdolgozatomban speciális eseteket vizsgáltam, melyre van hatékony, polinomiális időben lefutó algoritmus, mely megmondja a particionálás optimális méretét vagy magát a partíciót is. Először a $k = 2$ esetet tekintettem át, azaz a teljes párosítást, ezután a $k \geq 3$ esetén speciális gráfosztályokat vizsgáltam. Az első vizsgált gráfosztály a fák voltak, ahol lineáris időben lefutó algoritmus segítségével meghatározható a partíció részek minimális száma [5]. Utána a kaktuszok csoportjával foglalkoztam, ahol polinomiális időben futó algoritmust vizsgáltam, mely nem csak a paritició részek számát mondja meg, hanem a konkrét felosztást is, ennek eredményeképp kaktuszokon belül [6], a fákra is alkalmazható az algoritmus, mely a partíciókat is meghatározza. Az utolsó csoport a páros permutációs gráfoké volt, ahol szintén polinomiális időben lefutó algoritmust láthattunk, mely megmondja a minimális k-út particionálás méretét és a partíciókat egyaránt [2].

Ezek az eredeti üzleti problémára sajnos nem nyújtanak megoldást, mert abban általános gráfba rendeződnek a pontok, aminek k-út particionálása NP-nehéz probléma. A kiinduló problémára emiatt csak közelítő, de nem feltétlenül az optimális megoldást nyújtó heurisztikus algoritmusok vannak.

Irodalomjegyzék

[1] FRANK ANDRÁS, KIRÁLY TAMÁS *Operációkutatás*, Typotex kiadó (2014)

[2] GEORGE STEINER *On the k-path partition of graphs*, Theoretical Computer Science 290 (2003) 2147 – 2155

[3] GEORGE STEINER *On the k-path partition in cographs*, Congressus Numerantium 147 (2000) 89-95

[4] SZABÓ PÉTER GÁBOR, LONDON ANDRÁS, PLUHÁR ANDRÁS *Optimalizálási ismeretek*, (egyetemi jegyzet) 2018

[5] JING-HO YANG, GERARD J. CHANG, SANDRA M. HEDETNIEMI, STEPHEN T. HEDETNIEMI, *k-Path partitions in trees*, Discrete Applied Mathematics 78 (1996) 227-233

[6] ZEMIN JIN, XUELIANG LI, *On the k-path cover problem for cacti*, Theoretical Computer Science 355 (2006) 354-363

[7] SZESZLÉR DÁVID, *A DFS algoritmus*, BME Számítástudományi és Információelméleti Tanszék, (2015 – 2019) (http://cs.bme.hu/bsz2/dfs.pdf)