Eötvös Loránd Tudományegyetem

Természettudományi Kar

Tóth Bence Barnabás

Gráfelméleti megközelítések kódoló hálózatok műveletigényének becslésére

BSc Szakdolgozat

Témavezető:

Kovács Erika Renáta

Operációkutatási Tanszék

Budapest, 2013

Köszönetnyilvánítás

Ezúton szeretném megköszönni Kovács Erikának, hogy elvállalta a konzulensi teendőket. Köszönöm, hogy a konzultációk során türelmével, tudásával és szakmai tapasztalatával nagymértékben segítette munkámat, és irányított a szakdolgozat felépítésének megalkotásában.

Tartalomjegyzék

1. fejezet

Bevezetés

A kódoló hálózatok

Egy N kódoló hálózatban egy kijelölt forrásból kell h darab különböző információcsomagot eljuttatnunk k darab célpont mindegyikébe. A hálózat többi csúcsát innentől *belső csúcsoknak* fogjuk nevezni. A belső csúcsok két csoportra oszthatók, kódoló és továbbító csúcsokra. A kódoló csúcsok új információcsomagot készítenek több beérkezőből, a továbbítók pedig egy beérkezőt sokszorosítanak. A kódoló csúcsok számát szeretnénk minimalizálni a későbbiekben. Aciklikus hálózatban erre van egy csak h-tól és k-tól függő felső korlát, tehát N csúcsainak számától nem függ.

A kódoló hálózatok problémaköre megközelíthető gráfelméleti eszközökkel. Vegyünk egy aciklikus digráfot, melyben van egy s *forrás* csúcs, amibe nem vezet él, és $t_1, \ldots, t_k$ *célpont* csúcsok, melyekből nem vezet ki él. Ha az $s-t_i$ vágások minimuma h_i, akkor Menger tétele szerint van h_i éldiszjunkt út s-ből t_i-be, legyen $h = \min_{i=1..k}(h_i)$. Ezek az utak, ha azonos végponttal rendelkeznek, akkor éldiszjunktak, azonban ha különböző a végpontjuk, akkor lehet közös részútjuk. Ezen részutak kezdőpontjai a kódoló csúcsok. Ismert a kódoló csúcsok maximális számára adott h^2 nagyságrendű alsó, valamint h^3 nagyságrendű felső korlát, pontos érték azonban nem.

A szakdolgozat célja a kódoló csúcsok maximális számára adott korlátok ismertetése, valamint a kódoló csúcsok maximális számának meghatározására szolgáló algoritmusok és implementációik bemutatása. A problémát először korlátozzuk egyszerű hálózatokra, melyekben az utak csak egy féleképpen választhatók, és minden belső csúcs foka 3. Ezután az egyszerű hálózatok közül keresünk olyat, melyben a kódoló

csúcsok száma maximális. A dolgozatban két algoritmus szerepel. Az első módszer során felépítjük a gráfokat adott kódolócsúcs-szám esetén, majd ellenőrizzük, hogy megfelel-e az egyszerűségi feltételeknek. Az első algoritmus kevesebb gráfon iterál, de lassabb az ellenőrzése. A második algoritmus az elsőnél több esetet jár végig, de az ellenőrzést a gráfok építése közben végzi, így egyszerre egynél esetet is kizárhat, valamint gyorsabb az ellenőrzés szubrutinja is. A dolgozatban Langberg, Sprintson és Bruck eredményeit dolgoztam fel főként [1], és ennek terminológiájára építve fogalmaztam meg azon tételeket is, melyek Li, Weiping és Guangyue eredményeiben szerepelnek [2]. Az ábrák [1]-ből származnak, az algoritmusok implementálásakor a LEMON-t használtam [3]. Az 3.1 és 3.2 tételei és állításai [1]-ben, a 3.4 tétele a [2]-ben van leírva.

2. fejezet

A probléma megközelítése

2.1. Egyszerű hálózatok

A hálózat: $N(G, s, T, h)$, ahol G egy irányított, gyengén összefüggő gráf, melyben van egy s forrás, és k darab célpont csúcs, ezek halmaza T, valamint egy h természetes szám, ennyi információcsomagot kell eljuttatni a forrástól minden célpontig. Feltesszük továbbá, hogy G aciklikus, s-be nem vezet be él, T elemeiből pedig nem vezet ki él.

2.1.1. Definíció. *A hálózat* ***elfogadott****, ha a gráfjában s-ből minden célpontba vezet legalább h élfüggetlen út.*

A kódoló csúcsok definiálásához először rögzítsünk h éldiszjunkt utat a célpontok mindegyikéig, az hogy egy csúcs kódol-e ezen utaktól függ.

2.1.2. Definíció. *Legyen $N(G, s, T, h)$ elfogadott kódoló hálózat, melyben rögzítettünk h éldiszjunkt utat s-ből a célpontok mindegyikébe. Egy v belső csúcsot* ***kódoló csúcsnak*** *nevezünk, ha van olyan kimenő éle, melyet két rögzített út is úgy használ, hogy ezek más-más bejövő élet használnak. Tehát a kódoló csúcsok két rögzített út közös részútjainak kezdőpontjai.*

Egy u belső csúcsot ***továbbító csúcsnak*** *nevezünk, ha nem kódoló csúcs. Egy $N(G, s, T, h)$ hálózatban a kódoló csúcsok minimális számát jelöljük $C(N)$-nel, ez a minimum az utak összes lehetséges kiosztására vonatkozik. Innentől $C(N)$-et röviden a* ***kódoló csúcsok számának*** *fogjuk hívni.*

A kódoló csúcsok számának vizsgálatát leszűkítjük a hálózatok egy típusára, melyet egyszerű hálózatoknak fogunk nevezni. Belátjuk, hogy a kódoló csúcsok számának becsléséhez elég az egyszerű hálózatokat vizsgálni.

2.1.3. Definíció. ***Kanonikus hálózatnak*** *nevezzük azokat a hálózatokat, melyekben minden belső pont foka legfeljebb 3.*

2.1.4. Definíció. *Egy kódoló hálózat* ***minimális****, ha elfogadott, és tetszőleges élet elhagyva a gráfból már nem elfogadott.*

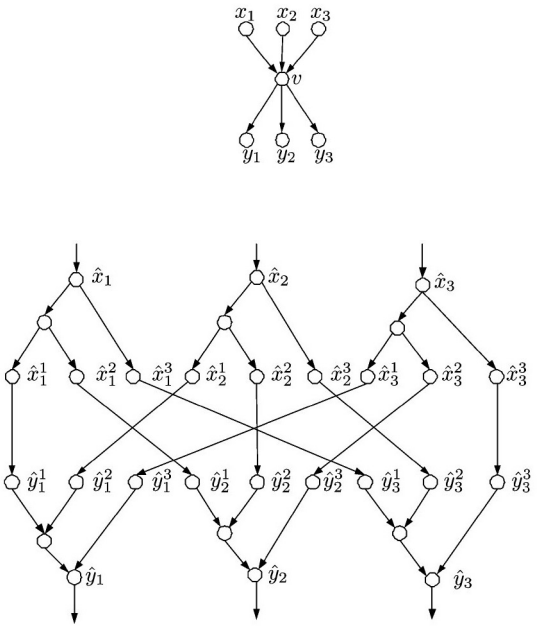
2.1.5. Definíció. ***Egyszerű hálózatnak*** *nevezzük azokat a minimális hálózatokat, melyekben minden belső pont foka legfeljebb 3. Tehát azokat a hálózatokat, melyek minimálisak és kanonikusak is.*

A következőkben egy eljárást mutatunk, mely minden $N(G, s, T, h)$ elfogadott kódoló hálózatot redukál egy $N'(G', s, T, h)$ egyszerű kódoló hálózattá. Az eljárás során először is minden belső pontot, melynek foka több, mint 3 helyettesítünk egy struktúrával. Egy v csúcsra legyen G_v a helyettesítő részgráf. Legyenek v bejövő és kimenő élei az alábbiak: $\{(x_i, v) \mid i = 1, \ldots, d_{in}(v)\}$ és $\{(v, y_i) \mid i = 1, \ldots, d_{out}(v)\}$, ahol $d_{in}(v)$ és $d_{out}(v)$ v be és kifokai. Minden egyes (x_i, v) bejövő élhez adjunk egy $\hat{x_i}$ csúcsot és egy X_i tetszőleges bináris fát, amelynek $\hat{x_i}$ a gyökere, a levelei pedig $\hat{x_i}^1, \ldots, \hat{x_i}^{d_{out}(v)}$. Hasonlóan minden (v, y_i) kimenő élhez adjunk egy $\hat{y_i}$ csúcsot és egy Y_i fordított bináris fát $\hat{y_i}$ gyökérrel, $\hat{y_i}^1, \ldots, \hat{y_i}^{d_{in}(v)}$ levelekkel. Ez után minden $1 \leq i \leq d_{in}(v)$ és $1 \leq j \leq d_{out}(v)$-re adjuk hozzá G_v-hez az $(\hat{x_i}^j, \hat{y_j}^i)$ élet. Végül G_v-t összekapcsoljuk a hálózat többi részével: minden $1 \leq i \leq d_{in}(v)$ és $1 \leq j \leq d_{out}(v)$-re adjuk az $(x_i, \hat{x_i})$ és $(\hat{y_i}, y_i)$ éleket G-hez.

Ebben a gráfban is van h éldiszjunkt út a forrásból minden célpontba, mert a G_v részgráfok konstrukciójából adódóan, minden $\hat{x_i}$-ből egyértelmű út vezet $\hat{y_j}$-be. Két ilyen út diszjunkt, ha különbözők a kezdő- és végpontjaik.

Most minimálissá tesszük N'-t. Ehhez mohó algoritmussal járjuk végig az éleket, majd nézzük meg, hogy a hálózat elfogadott-e ha az adott élet kitöröljük. Ha igen, hagyjuk el, ha nem, akkor maradjon a hálózatban. Az esetleg keletkező izolált csúcsokat szintén elhagyjuk.

2.1.6. Megjegyzés. Ebben az új hálózatban szintén van h élfüggetlen út a célpontokig, mivel a szétosztásnál a konstrukció ezt megtartja, az élek elhagyásánál pedig ha nem lenne, akkor nem hagyjuk el az élet.



2.1.7. Állítás. *Egy minimális hálózat esetén a kódoló csúcsok $C(N)$ száma legfeljebb akkora, mint a belőle készített egyszerű hálózatban.*

Bizonyítás. Az eljárás során minden v belső csúcsot helyettesítettünk a gráfban, egy G_v részgráffal. Most N' minden útkiosztásához, mely l kódoló csúcsot indukál, készítünk egy útkiosztást N-ben, legfeljebb ennyi kódoló csúccsal, ezzel belátva, hogy $C(N) \leq C(N')$. Legyen P egy (s, t_i) út, valamelyik célpontba N'-ben. Nézzük, hogy ez a G_v részgráfok között milyen éleket használ, ezek megfelelőin vezessük az utat N-ben. N-ben pontosan azok a v csúcsok lesznek kódoló csúcsok, amelyekre G_v tartalmazott kódoló csúcsot. Ezek kódolni fognak, mert itt találkoznak utak, melyek együtt mennek tovább, valamint más csúcs nem lehet kódoló, mert ha két út találkozott v-ben és közös élen haladtak tovább, akkor az N'-ben ennek megfelelő G_v-ben is kellett lennie kódoló csúcsnak. □

2.1.8. Következmény. Amennyiben a minimális kódoló hálózatokban szeretnénk vizsgálni a kódoló csúcsok maximális számát, elég az egyszerű hálózatokat vizsgálni.

2.1.9. Megjegyzés. Egy egyszerű hálózatban pontosan azok a csúcsok kódoló csúcsok, melyek befoka 2, kifoka 1, mivel a hálózat minimális, így mindkét bejövő élet használnia kell egy útnak, és ezek egyazon élen mennek tovább.

3. fejezet

Elméleti eredmények

3.1. Felső korlát a kódoló csúcsokra

A $k = 2$ esettel foglalkozunk elsőként. Legyen $N(G, s, T, h)$ egy kanonikus hálózat (ettől még N nem feltétlenül egyszerű), melyben 2 célpont csúcs van. A két célpontot nevezzük t_1 és t_2-nek. Rögzítsünk h élfüggetlen utat s-ből t_1-be, ezeket innentől kezdve *piros utaknak* fogjuk hívni. Hasonlóan rögzítsünk h utat s-ből t_2-be, ezek lesznek a *kék utak*. Egy élet hívjunk piros élnek, ha eleme egy piros útak, kék élnek, ha eleme egy kék útnak. Egy él lehet piros és kék úton is, ekkor a színe piros és kék egyszerre.

Az összes t_1-be belépő él kizárólag piros, és az összes t_2-be belépő él kizárólag kék, mert a célpontokból nem vezet ki él, és a piros utak végpontja t_1, a kékeké pedig t_2. Amennyiben a hálózat minimális, az alábbi tulajdonságok is fennállnak: a hálózatban minden él rendelkezik legalább az egyik színnel, és minden v belső pontra igaz, hogy egy adott színből legfeljebb egy út használja, mivel minden ilyen pont foka maximum 3.

Most két segédgráfot fogunk megadni, melyek akkor és csak akkor aciklikusak, ha $N(G, s, T, h)$ minimális. Ezen gráfok függenek a piros és kék utak megválasztásától.

3.1.1. Definíció. *Legyen az $N(G, s, T, h)$ kódoló hálózathoz tartozó **első segédgráf** $G_1(N)$. Ezt G-ből kapjuk az összes piros él megfordításával. Tehát azok az élek, amelyek egyszerre kékek és pirosak is, megfordulnak.*

3.1.2. Definíció. *Legyen az $N(G, s, T, h)$ hálózathoz tartozó **második segédgráf** $G_2(N)$. Ezt G-ből kapjuk az összes kék él megfordításával.*

3.1.3. Tétel. *Ha N egyszerű (tehát minimális), akkor $G_1(N)$ és $G_2(N)$ aciklikusak (így $G_2\hat{(}N)$ is aciklikus).*

Bizonyítás. Jelöljük a h darab piros (s, t_1) utat $P_1, \ldots, P_h$-val, a kék (s, t_2) utakat pedig $\hat{P}_1, \ldots, \hat{P}_h$-pal. Mivel N egyszerű, és így minimális, G minden éle színezett legalább egy színnel. Indirekt tegyük fel, hogy van egy C kör $G_1(N)$-ben. Minden $e \in C$ élre van egy e' él G-ben, ami a megfelelő csúcsok között fut, legfeljebb az iránya fordított. A továbbiakban e színén e' színét értjük.

Először megfigyeljük, hogy C csak belső pontokat tartalmaz, mert a forrásnak csak kimenő, a célpontoknak csak bejövő éleik vannak.

Másodszor: C-ben biztosan van olyan e_1 él, amely kizárólag kék színű, különben G-ben lenne egy kör a piros élek között, de G aciklikus volt.

Harmadszor: C-ben biztosan van egy kizárólag piros e_2 él. Ez azért van, mert ha nem lenne ilyen, akkor az összes él C ben kék vagy kék-piros lenne. Ekkor legalább az egyik, például e_1 is, tisztán kék, vagyis ennek az iránya ugyanaz G-ben és C-ben. C-ben van kék-piros él is, mert ha ez nem igaz, akkor G-ben van egy kör csupa kék élekből, de G aciklikus. Ennek a kék-piros élnek az iránya ellenkező G-ben és C-ben. Az előzőek miatt van 2 él C-ben, (u, v) és (v, w), melyek közül (u, v) csak kék, (v, w) pedig kék-piros. Az eredeti G gráfban tehát a v csúcsnak két kék bejövő éle van, de v belső pont, ezért ekkor a kifoka is legalább 2. Ez ellentmond a feltételezésnek, mely szerint N egyszerű, tehát van egy kizárólag piros e_2 él C-ben.

Megmutatjuk, hogy e_2' elhagyható az eredeti G gráfból, ami ellentmond a minimalitásának. Ehhez megadunk h éldiszjunkt utat s és t_1 között, melyek nem tartalmazzák e_2'-t. Tekintsük a piros utakhoz tartozó élek $E' = \{e : e \in P_i, i = 1, \ldots, h\}$ halmazát. Az E' halmaz egy h értékű folyamot határoz meg s ből t_1-be a G-gráfban. Tegyük még fel, hogy e_2' benne van E'-ben, különben elhagyva megmarad a h-folyam. Készítsük el az E'' folyamot melyet C felhasználásával kapunk E'-ből. E'' élei az alábbiak lesznek: minden él, mely benne van E'-ben és a fordítottja nincs benne C-ben, valamint minden él, mely benne van C-ben és a fordítottja nincs benne E'-ben.

$$E'' = \{(v, u) : (v, u) \in E' \text{ és } (u, v) \notin C\} \cup \{(v, u) : (v, u) \in C \text{ és } (u, v) \notin E'\}$$

E'' nem tartalmazza e_2'-t, mert e_2 benne volt C-ben. Be kell még látnunk, hogy minden (V_1, V_2) vágás, mely s-et és t_1-et elválasztja, legalább h élet tartalmaz E''-ből. Ez igaz, mert E' tartalmazott h élet, és C pedig ugyanannyi élet tartalmaz

az egyik irányban V_1 és V_2 között, mint fordított irányban. Adjunk E' éleinek 1 értéket, a többinek 0-t. Ezután adjunk C minden élének 1 értéket, és adjuk össze a két folyamot, úgy hogy a szembe menő élek kioltják egymást. Ekkor megkapjuk E''-t, a folyam értéke pedig nem változik. Tehát adtunk egy új folyamot, mely nem tartalmazza e_2-t

Ezt a bizonyítást alkalmazhatjuk $G_2(N)$ fordítottjára is, ezzel az állítást beláttuk. □

3.1.4. Következmény. Ha $N(G, s, T, h)$ egyszerű hálózat, a piros és kék utak, így a segédgráfok is egyértelműen meghatározottak.

3.1.5. Tétel. *Ha N kanonikus, $G_1(N)$ és $G_2(N)$ aciklikusak, és minden élen át vezet út legalább az egyik célpontba, akkor N minimális.*

Bizonyítás. A bizonyítás indirekt. Tegyük fel, hogy N nem minimális, ekkor egy e él elhagyásával még mindkét célpontba vezet h élfüggetlen út. Használjuk, hogy e-n át vezet út az egyik célpontba, mondjuk t_1-be. A piros utak (ezek vezetnek t_1-be) meghatározásánál válasszuk úgy az utakat, hogy e ne legyen rajta egyik úton se. Legyen U egy olyan (s, t_1) út, mely tartalmazza e-t. Ekkor $G_1(N)$-ben lesz kör, mert G-ben volt két (s, t_1) út (az egyik U, a másik egy piros út) és ezek közül az egyiket megfordítottuk. Mivel nem minden élben egyeztek, de a kezdő és végpontjuk ugyanaz, így lesz valahol egy kör. Ha e-n át t_2-be vezet út, akkor $G_2(N)$ fog kört tartalmazni. □

A felső korlát megtalálásához be kell vezetnünk még egy $\hat{G}$ segédgráfot, és pár új utat, melyek *zöldek* lesznek.

3.1.6. Definíció. *Legyen $N(G, s, T, h)$ egy egyszerű hálózat, két célponttal, a piros utak $\{P_1, \ldots, P_h\}$, a kékek pedig $\left\{\hat{P}_1, \ldots, \hat{P}_h\right\}$. A G-hez tartozó $\hat{G}$ segédgráfot G-ből készítjük, úgy hogy elhagyjuk azon éleket, melyek kékek és pirosak is egyszerre, majd megfordítjuk a piros éleket.*

3.1.7. Megjegyzés. $\hat{G}$ részgráfja $G_1(N)$-nek és $G_2(N)$-nek is.

Ahogy egyszerű N esetén $G_1(N)$ és $G_2(N)$ definíciója, $\hat{G}$ definíciója is egyértelműen adódik a gráfból, mivel csak egy féleképpen választhatjuk a piros és kék utakat. A következőkben belátjuk, hogy $\hat{G}$ h élfüggetlen (t_1, t_2) útból áll. Ezeket fogjuk *zöld utaknak* nevezni.

3.1.8. Állítás. *Egy N egyszerű hálózathoz tartozó $\hat{G}$ segédgráf h élfüggetlen (t_1, t_2) útból épül fel ($\hat{G}$-ben természetesen lehetnek még izolált csúcsok).*

Bizonyítás. Először belátjuk, hogy minden $v \in \hat{G}$ belső csúcsra v-nek pontosan egy bejövő és egy kimenő éle van (vagy nincs éle). Minden belső csúcsra az alábbi esetek egyike áll fenn az eredeti G gráfban:

1. v-nek egy e bejövő, és egy e' kimenő éle van. Ekkor mindkettő csak piros, mindkettő csak kék, vagy mindkettő kék-piros. Az első két esetben mindketten benne vannak $\hat{G}$-ben (ha pirosak voltak, akkor fordítva), a harmadik esetben egyikük sincs.

2. v-nek két bejövő éle van, e_1 és e_2, valamint egy e' kimenő éle. Ebben az esetben egy kék és egy piros út lép be v-be két különböző bejövő élen, és együtt mennek ki e'-n. Olyan él nem lehet melynek nincs színe. Ekkor e' nem lesz benne $\hat{G}$-ben, a bejövő élek közül a piros megfordul, így v-nek egy bejövő és egy kimenő éle lesz $\hat{G}$-ben.

3. v-nek egy e bejövő éle van, valamint két kimenő éle, e'_1 és e'_2. Most egy kék és egy piros út lép be együtt e-n és külön mennek ki e'_1 és e'_2-n. $\hat{G}$ nem fogja tartalmazni e-t, e'_1 és e'_2 közül a piros megfordul, így v-nek egy bejövő és egy kimenő éle lesz $\hat{G}$-ben.

Most megmutatjuk, hogy a forrás ki- és befoka $\hat{G}$-ben ugyanannyi. Mivel N minimális, minden élnek van színe, így s-ből h piros és h kék él lép ki, be egy sem. Azok az élek, melyek kékek és pirosak is egyszerre, kitörlődnek. Ez után ugyanannyi piros marad, mint kék. A pirosak megfordulnak, tehát tényleg egyenlő s be- és kifoka.

A célpontok közül t_1-nek 0 a befoka és h a kifoka, t_2-nek pedig h a befoka és 0 a kifoka, mert G-ben t_1-be csak piros, t_2-be csak kék él lépett be.

Összegzés képpen $\hat{G}$ a célpontokon kívül csupa olyan pontból áll, melynek befoka egyenlő a kifokával, valamint tudjuk, hogy $\hat{G}$ aciklikus, mert $\hat{G}$ részgráfja $G_1(N)$-nek, és mivel $G_1(N)$ aciklikus, $\hat{G}$ is az. Ebből következően van h éldiszjunkt út t_1-ből t_2-be, nevezzük ezeket zöld utaknak. Végül: $\hat{G}$-ben minden él része zöld útnak, mert ha nem lenne, a zöld éleket először elhagyva, egy ilyen maradék élből kiindulva építünk egy sétát, mely vagy kört fog tartalmazni, vagy elér t_2-be. Kör nem lehet, t_2-be pedig nem juthatunk, mert annak pontosan h volt a befoka. □

3.1.9. Állítás. *Legyen $N(G, s, T, h)$ egy egyszerű hálózat. Ekkor G minden $e = (v, u) \in G$ élére az alábbi tulajdonságok közül pontosan az egyik fordul elő:*

a) A G gráfban e piros és kék egyszerre.

b) e színe piros és zöld. Ekkor G-ben piros de nem kék volt, és a fordítottja $(u, v) \in \hat{G}$.

c) e színe kék és zöld. Ekkor e G-ben kék volt, de piros nem, és $e \in \hat{G}$, ezért zöld.

Bizonyítás. Egy e élre sorra nézzük, hogy G-ben milyen színe lehetett, miután az utakat kiosztottuk.

1. Az e él egy piros és egy kék úthoz is tartozik G-ben. Ekkor $e \notin \hat{G}$, és így *a* eset fenn áll, mivel $e \notin \hat{G}$, így nem lehet zöld.

2. Az e él kizárólag piros. Most a fordítottja benne van $\hat{G}$-ben, ezért zöld is, a *b* eset fordul elő, a másik kettő nem, mert e nem kék.

3. Az e él kizárólag kék. Ekkor benne van $\hat{G}$-ben, tehát a *c* eset áll fenn, a másik kettő nem, mivel ezek megkövetelik, hogy e piros is legyen.

□

Eredmény képpen minden él pontosan 2 színnel rendelkezik a kék, piros és zöld színek közül. Most már készen állunk a két célpontos egyszerű hálózatokban levő kódoló csúcsok felső korlátjának megadására. Ehhez egy állítást kell belátni még.

3.1.10. Állítás. *Legyen $N(G, s, T, h)$ egy egyszerű hálózat, két célpont csúccsal. Rögzítsünk h élfüggetlen (s, t_1) piros utat, ezek halmaza legyen S_p, valamint h élfüggetlen (s, t_2) kék utat, ezek halmaza legyen S_k. Legyen $\hat{G}$ a fent definiált segédgráf h élfüggetlen (t_1, t_2) zöld úttal, melyek halmazát jelöljük S_z-vel. Legyen $P_p \in S_p$ egy piros út, $P_k \in S_k$ egy kék út és $P_z \in S_z$ egy zöld út. Ekkor maximum egy olyan $v \in G$ csúcs van, ami egyaránt rajta van P_p-n, P_k-n és P_z-n is.*

Bizonyítás. Készítsük el az N-hez tartozó első és második segédgráfokat, $G_1(N)$ és $G_2(N)$-et. A zöld P_z út benne van mindkét segédgráfban. Legyen $P'_p \in G_1$ az a (t_1, s) út, amit a kiválasztott piros út megfordításával kapunk. Indirekt tegyük fel, hogy létezik két csúcs, mely rajta van P_p-n, P_k-n és P_z-n is. Jelöljük ezeket a csúcsokat v_1

és v_2-vel, mégpedig úgy, hogy P_z-n v_1 előbb jön, mint v_2. Ekkor v_1 megelőzi v_2-t a P_k úton is, különben $G_2(N)$, mely tartalmazza P_k-t és P_z-t is nem lehetne aciklikus, ez pedig ellentmond N minimalitásának. A P_p úton v_2 előbb jön, mint v_1 (így P'_p-n v_1 megelőzi v_2-t), mert ha ez nem igaz, $G_1(N)$, mely ben benne van P'_p és P_z, kört fog tartalmazni, mely ismét ellentmondást okoz N minimalitásával. Tehát v_1 megelőzi v_2-t P_k-n, de mögötte van P_p-n. Emiatt lenne egy kör G-ben, ez azonban ellentmond G aciklikusságának. □

A következő tétel mondható ezek felhasználásával:

3.1.11. Tétel. *Legyen $N(G, s, T, h)$ aciklikus, egyszerű hálózat, ahol két célpont csúcs van. Ekkor a kódoló csúcsok $C(N)$ számára igaz a következő: $C(N) \leq h^3$.*

Bizonyítás. Rögzítsünk G-ben h piros (s, t_1) utat, jelöljük ezek halmazát S_p-vel, valamint h kék (s, t_2) utat, jelöljük ezek halmazát S_k-val. Készítsük el a $G_1(N)$, $G_2(N)$ és $\hat{G}$ segédgráfokat, legyen S_g a zöld (t_1, t_2) utak halmaza $\hat{G}$-ben. Jelöljük V'-vel a belső pontok azon részhalmazát, ahol a pontok befoka 2. Ekkor V' pontosan a kódoló csúcsokat tartalmazza, így $|V'| = C(N)$. Minden $v \in V'$ csúcsnak két bejövő éle van, az egyik piros, a másik kék, és egy kimenő éle, mely piros és kék is. A 3.1.9 állítás következménye képpen v rajta van egy $P_p \in S_p$ piros, egy $P_k \in S_k$ kék és egy $P_z \in S_z$ zöld úton. A 3.1.10 állítás miatt 3 különböző színű úthoz egyszerre maximum egy csúcs tartozhat. Mivel pontosan h piros út, h kék út és h zöld út van, ezért V' elemszáma, így a kódoló csúcsok száma maximum h^3. □

Most megfogalmazunk egy állítást a $k > 2$ esetre, ahol $k = |T|$ a célpont csúcsok száma.

3.1.12. Tétel. *Legyen $N(G, s, T, h)$ aciklikus, egyszerű hálózat, melyben k a célpontok száma. Ekkor $C(N) \leq h^3k^2$.*

Bizonyítás. Kezdjük a következő észrevétellel. Legyen $t_i, t_j \in T$ két célpont, ekkor van h éldiszjunkt út s-ből mindkettőbe. Legyen $G'_{(i,j)}$ az ezen utak által meghatározott részgráf. Erre $N'(G'_{(i,j)}, s, \{t_1, t_2\}, h)$ elfogadott, és kanonikus is, de nem feltétlenül minimális. Készítsük el ebből az $N_{(i,j)}(G_{(i,j)}, s, \{t_1, t_2\}, h)$ egyszerű hálózatot az elhagyható élek kitörlésével. Jelöljük $G_{(i,j)}$-ben a h darab élfüggetlen, s-ből t_i-be és t_j-be vezető utak halmazait $S^i_{(i,j)}$, valamint $S^j_{(i,j)}$-vel. Legyen a $G_{(i,j)}$ gráfban a kódoló csúcsok halmaza $V_{(i,j)}$. A 3.1.11 tétel szerint $C(N_{(i,j)}) \leq h^3$.

Indirekt tegyük fel, hogy $C(N) > h^3k^2$. Ekkor van legalább egy v kódoló csúcs, mely nincs benne az $\bigcup_{t_i,t_j \in T} V_{(i,j)}$ halmazban. Jelöljük e' és e''-vel a két v-be bejövő élet, e-vel a v-ből kimenő élet. Megmutatjuk, hogy e' vagy e'' elhagyható $N(G,s,T,h)$-ból. Minden t_i célpontra van néhány $\left\{S^i_{(i,j)} \mid t_j \in T\right\}$ halmaza a h élfüggetlen (s,t) utaknak. Ha minden $t_i \in T$ célpontra létezik olyan $S^i_{(i,j)}$, mely nem tartalmazza e'-t, akkor e' elhagyható a hálózatból, az elfogadottság megsértése nélkül, ekkor N nem lehetett minimális. Tehát kell lennie egy t_i célpontnak, melyre e' benne van minden lehetséges $\left\{S^i_{(i,j)} \mid t_j \in T\right\}$-ben. Ekkor azonban tetszőlegesen választott $\left\{S^j_{(i,j)} \mid t_j \in T\right\}$ nem tartalmazhatja e''-t, így azonban sérül N minimalitása. Ha pedig tartalmazná, akkor e'' benne volna valamilyen $S^j_{(i,j)}$-ben, ekkor $v \in V_{(i,j)}$, ez is ellentmondásra vezet. □

3.2. Alsó korlát a kódoló csúcsok maximális számára

Az előző részben mutattunk egy felső korlátot a kódoló csúcsok számára. A végső cél a pontos érték megtalálása, most szeretnénk egy alsó korlátot találni. A következőkben egy konstrukciót fogunk adni olyan $N(G,s,T,h)$-ra, melyben $k = 2$ esetén a kódoló csúcsok száma $\frac{h(h-1)}{2}$.

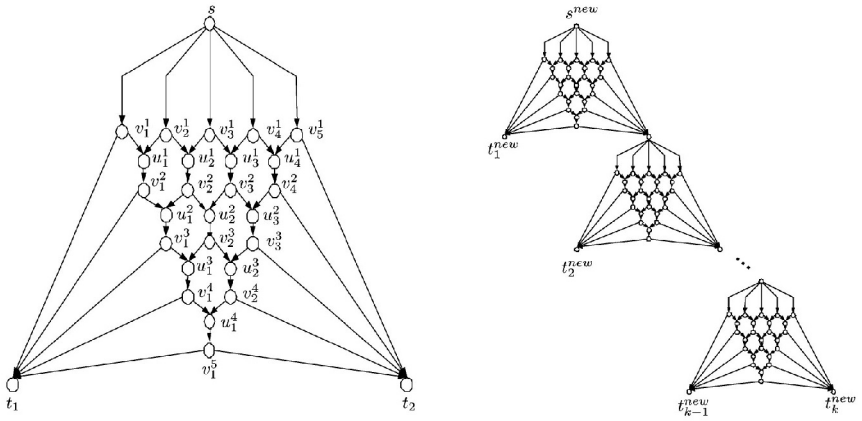
3.2.1. Állítás. *Legyen $h > 2$ egész szám. Ekkor létezik olyan $N(G,s,T,h)$ kódoló hálózat 2 célpont csúccsal, melyre a kódoló csúcsok száma $\frac{h(h-1)}{2}$.*

Bizonyítás. Készítsük a hálózat gráfját az alábbi módon. Egyszerű hálózatot fogunk készíteni, vagyis minden belső pont fokszáma maximum 3 (a konstrukcióban pontosan 3). Legyenek az s-ből kiinduló h él különböző végpontjai a $v^1_1, v^1_2, \ldots, v^1_h$ csúcsok. Ezek nem kódoló csúcsok, csak továbbítók. Legyenek további csúcsok még $u^1_1, \ldots, u^1_{h-1}$. Élek futnak v^1_1-ből t_1-be és v^1_h-ből t_2-be, valamint minden $0 \leq i \leq h-1$-re v^1_i-ből és v^1_{i+1}-ből u^1_i-be. Az első szintünk kész, a többi hasonlóan épül fel. Ezek után a $h-1$ darab u^1 csúcsból kiinduló $h-1$ élnek hozzunk létre új végpontokat, $v^2_1, v^2_2, \ldots, v^2_{h-1}$-t. Innen a konstrukció ugyanaz mint az előbb volt, új csúcsok $u^2_1, \ldots, u^2_{h-2}$, új élek (v^2_1, t_1) és (v^2_{h-1}, t_2) valamint minden $0 \leq i \leq h-2$-re (v^2_i, u^2_i) és (v^2_{i+1}, u^2_i). Tetszőleges szintre csúcsok voltak eddig $v^j_1, v^j_2, \ldots, v^j_{h-j+1}$, új

csúcsok $u_1^j, \ldots, u_{h-j}^j$. Az élek a következők: (v_1^j, t_1) és (v_{h-j+1}^j, t_2), valamint minden $0 \leq i \leq h-j$-re (v_i^j, u_i^j) és (v_{i+1}^j, u_i^j). Ez után még a gráfhoz hozzávesszük a $v_1^{j+1}, \ldots, v_{h-j}^{j+1}$ csúcsokat és az $(u_1^j, v_1^{j+1}), \ldots, (u_{h-j}^j, v_{h-j}^{j+1})$ éleket. Ezt csináljuk, amíg $j \leq h-1$, majd az utolsó fennmaradó v_1^h csúcsot összekötjük mindkét célponttal.

Nézzük most, hány kódoló csúcsunk van! A hálózatban pontosan az u csúcsok lesznek a kódoló csúcsok, ezek száma $\frac{h(h-1)}{2}$. Mindkét célpontba pontosan h út vezet, és ezek éldiszjunktak. □

Ehhez a példához hasonlóan lehet megadni $k > 2$-re olyan hálózatot, melynek $(k-1) \cdot \frac{h(h-1)}{2}$ kódoló csúcsa van. Ezt úgy kapjuk, hogy felépítjük az előző példát s kezdőpont, t_1 és t_2' célpontokba, ahol t_2' egy új csúcs, amit hozzáadunk a gráfhoz. Ezután t_2' kezdőponttal építünk t_2 és t_3' célpontokba hasonlóan. Ezt addig folytatjuk, amíg el nem érünk t_{k-1} és t_k célpontokig, itt befejezzük. Ezzel az előző példát $k-1$-szer építettük fel, így a kódoló csúcsok száma $(k-1) \cdot \frac{h(h-1)}{2}$. Az ábrán a $h = 5$ eset látható.



3.3. Az egyszerű hálózatok felépítése

Most néhány állítás következik az egyszerű hálózatok felépítéséről, innentől kezdve kizárólag a $k = |T| = 2$ esettel foglalkozunk. Az egyszerű hálózatokban minden belső pont foka legfeljebb 3. A minimalitás miatt 1 fokú csúcs nem lehet. Először is tegyük fel, hogy nincs 2 fokú csúcs. Ezek mindegyikének be és kifoka is 1, különben a hálózat nem lehetne minimális. Ezek nem lehetnek kódoló csúcsok, mert ott több útnak kell összetalálkozni, ez 1 befok esetén nem lehetséges.

3.3.1. Állítás. *Ha $N(G, s, T, h)$ egyszerű hálózat és s kifoka nagyobb h-nál, akkor vezet él s és t_1, valamint s és t_2 között.*

Bizonyítás. Indirekt bizonyítunk, tegyük fel, hogy s kifoka nagyobb h-nál, és nincs (s, t_1) él (az állítás t_2-re analóg módon bizonyítható). Ekkor van olyan (s, v^*) él, mely kizárólag piros, vagyis csak t_1-be vezető út használja, jelöljük P-vel ezt az utat. Most nézzük a v^* csúcsot! Ez belső csúcs a feltétel miatt, és mivel eltüntettük a 2 fokúakat, ezért 3 a foka. Két eset lehetséges.

1. v^* befoka 1, kifoka 2. Ekkor P a kimenő élek közül egyet használ, a másikat piros út nem használhatja, de kék sem, mert v^*-ba nem lép be kék él. Tehát a másik, P által nem használt kimenő él elhagyható a gráfból, ez ellentmond N minimalitásának.

2. v^* befoka 2, kifoka 1. Most két bejövő él van, (s, v^*) és e. Utóbbi színe nem piros, így ha N minimális, akkor kék. Hívjuk P'-nek a kék utat, ami használja e-t. Most a P' út v^* előtti részét kicserélhetjük az (s, v^*) élre, mely eddig nem volt kék, így e-t elhagyhatjuk a gráfból, ez pedig sérti N minimaitását.

□

Feltesszük ezentúl, hogy s kifoka h. Ha ennél több, mondjuk $h + l$, akkor van l darab s-ből kivezető kizárólag piros él, és szintén l darab s-ből kivezető kizárólag kék él. Ezek az élek az előző állítás következménye képpen a célpontokba vezetnek, így a kódoló csúcsok számlálásakor érdektelenek.

3.3.2. Állítás. *Ha $N(G, s, T, h)$ egyszerű hálózat és s kifoka h, akkor semelyik (s, v) élre sem lehet v kódoló csúcs, ezért a befoka 1, a kifoka pedig 2.*

Bizonyítás. Az (s, v) él színe most egyszerre piros és kék, ha v befoka 2 lenne, a másik élen nem jönne út, ez sértené a minimalitást. Mivel v belső pont, így a foka 3, a befoka 1, tehát a kifoka 2, azaz továbbító csúcs. □

3.3.3. Következmény. A kifokok összege a belső pontokra h-val több, mint a befokoké, ezért az (s, v) élek végpontjait nem számolva ugyanannyi kódoló csúcs van, mint továbbító.

Az (s, v) élek végpontjairól már mondtunk valamit, most következzen egy állítás az (u, t_i) élek kezdőpontjairól.

3.3.4. Állítás. *Ha $N(G, s, T, h)$ egyszerű hálózat, az (u, t_i) élek u kezdőpontjai nem lehetnek kódoló csúcsok.*

Bizonyítás. Az (u, t_i) élek egyszínűek, mert célpontba vezetnek, ezért ha u-nak két bejövő éle lenne az egyik színtelen, ami nem lehet. □

A gráf belső felépítésére igaz a következő:

3.3.5. Állítás. *Legyen $N(G, s, T, h)$ egyszerű hálózat, ebben v egy kódoló csúcs, tehát a befoka 2, a kifoka 1. Legyenek az u-hoz tartozó élek (u_1, v),(u_2, v) és (v, w). Ekkor u_1,u_2 és w továbbító csúcsok, vagyis 1 a befokuk és 2 a kifokuk.*

Igaz fordítva is, tetszőleges v továbbító csúcsnak nem lehet szomszédja másik továbbító csúcs.

Bizonyítás. Indirekt bizonyítunk, tegyük fel, hogy van egy v kódoló csúcsunk, melyből él vezet egy w kódoló csúcsba. Most tekintsük azt a $G^* \subset G$ részrgáfot, amit az u és w pontok határoznak meg. G-ben G^*-ból kivezet 1 él, és bevezet 3. A kimenő élet összesen maximum 2 út használja, egy piros és egy kék. G' be nem léphet be 2-nél több út, de három bejövő éle van, így az egyik elhagyható, ez pedig nem lehet, mert N minimális.

Ugyanígy bizonyítható az állítás továbbító csúcsokra, itt a részgráfba belépő utak száma maximum 2, a kilépő élek száma 3. □

Még két állítás következik, ezek a forrás és a célpontok távolságáról szólnak.

3.3.6. Állítás. *Legyen $N(G, s, T, h)$ egyszerű hálózat, ahol nincs 2 fokú belső csúcs. Ekkor van legfeljebb 2 hosszú út s-ből t_1-be és t_2-be.*

Bizonyítás. Tegyük fel, hogy s-ből h él vezet ki, tehát kizárjuk azokat az eseteket, amikor van él a kérdéses csúcsok között. Indirekt bizonyítunk, tegyük fel, hogy minden (s, t_1) és (s, t_2) út legalább 2 csúcsot tartalmaz a végpontokat nem számolva. Be fogjuk látni, hogy a $G_1(N)$ segédgráf tartalmaz kört, ami ellentmond N minimalitásának. Nézzük tehát a $G_1(N)$ gráfot, ebben a piros élek fordítva vannak. Egy algoritmust adunk kör keresésére. Induljunk ki t_1-ből, és válasszuk ki az egyik piros utat. Ezen menjünk el az s előtti utolsó csúcsig, jelöljük ezt v_1-gyel. Ez valamelyik (s, t_2) kék út első csúcsa, és a feltétel alapján a következő éle még nem t_2-be vezet. Ennek az élnek a végpontja tehát rajta van egy piros úton. Ezen most ismét menjünk el az s előtti utolsó, v_2 csúcsig. A többi lépés analóg módon következik,

addig megyünk, amíg az egyik v_i csúcsot nem érintjük másodszor. Itt biztosan kört kapunk $G_1(N)$-ben. □

3.3.7. Állítás. *Legyen $N(G, s, T, h)$ egyszerű hálózat, ahol nincs 2 fokú belső csúcs. Ekkor létezik olyan továbbító csúcs, melynek egyik szomszédja t_1, a másik t_2.*

Bizonyítás. Indirekt bizonyítunk, tegyük fel, hogy nem létezik ilyen csúcs. Belátjuk, hogy ekkor G nem lehet aciklikus. Induljunk ki s-ből, és válasszunk egy tetszőleges piros utat. Ezen menjünk végig a t_1 előtti utolsó, v_1 csúcsig. Ez továbbító csúcs, és a másik élén egy kék út folyatódik, mely feltevés szerint nem ér véget rögtön, ezen fogunk továbbmenni. Ismételjük az eljárást, amíg el nem jutunk egy olyan v_i csúcsba ahol már jártunk, így kaptunk egy kört G-ben. □

3.4. A felső korlát élesítése

A következőkben javítani szeretnénk a kódoló csúcsok számára adott becslést, ehhez felhasználjuk az előző részben leírt tulajdonságokat. Most kizárólag a $k = |T| = 2$ esettel fogunk foglalkozni.

3.4.1. Tétel. *Legyen $N(G, s, T, h)$ egyszerű hálózat, ekkor a kódoló csúcsok száma N-ben legfeljebb $\lceil \frac{h}{2} \rceil (h^2 - 4h + 5)$.*

Bizonyítás. A bizonyításhoz először nézzük az utakat, jelöljük az s-ből t_1-be vezető, piros utakat $P_1 \dots P_h$-val, a t_2-be vezető, kék utakat $Q_1 \dots Q_h$-val. Tegyük fel, hogy s-ből h él indul ki, az utak legyenek oly módon számozva, hogy P_i és Q_i induljon azonos élből minden i-re. A 3.3.6 állításnál láttuk, hogy van olyan piros és kék út is, mely mindössze 2 élből áll, legyenek ezek P_1 és Q_h. Azt mondjuk, hogy egy (P_i, Q_j) útpár *egyező*, ha $i = j$, *nem egyező*, ha $i \neq j$. Egy útpár *találkozó*, ha van olyan kódoló csúcs, ahol találkoznak. N-ben legfeljebb $(h - 2)$ egyező útpár van, ami találkozó is (a két rövid út különböző helyről indul, és ezek nem találkoznak senkivel). Legfeljebb $(h^2 - 3h + 3)$ nem egyező útpár van, mert a nem egyező párok száma $n(n - 1) = n^2 - n$, ebből le kell vonni $2(n - 1)$-et, mert a két rövid nem találkozhat egyik másikkal sem, majd hozzáadni 1-et, mert kétszer vontuk le, mikor a két rövid találkozik.

Ezután készítsük el a 3.1.6-ben definiált $\hat{G}$ segédgráfot és a zöld utakat, ezek közt mindig van legalább egy, ami 2 hosszú, ennek egy belső csúcsa van, ami továbbító

csúcs. A zöld utak minden belső csúcsához tartozik egy útpár, akár kódoló, akár továbbító csúcsról van szó. A következő megfigyelést tesszük: minden útpár maximum egyszer szerepel egy adott zöld út csúcsain. Ez azért igaz, mert ha nem állna fenn, akkor $\hat{G}$ nem lehet aciklikus, de következik a 3.1.10 állításból is. Igaz továbbá, hogy minden kék/piros út kezdés egyező útpárhoz tartozik, de ez nem kódoló csúcs.

Innen két eset lehetséges, ezek nem zárják ki egymást:

1. Van egy 2 hosszú zöld út, melynek belső csúcsához egyező útpár tartozik. Ehhez az útpárhoz maximum $\left\lfloor \frac{h-1}{2} \right\rfloor$ kódoló csúcs tartozik, mert minden zöld úthoz csak egy tartozhat, és a csúcsok fele kódol. Tetszőleges másik egyező párhoz legfeljebb $\left\lfloor \frac{h-2}{2} \right\rfloor$ kódoló csúcs lehet, itt az előző zöld utat kizárhatjuk. A nem egyező párok mindegyikéhez $\left\lfloor \frac{h-1}{2} \right\rfloor$ kódoló csúcs tartozhat maximum. A kódoló csúcsok számát felülről becsüli
$$\left\lfloor \frac{h-1}{2} \right\rfloor + (h-3) \left\lfloor \frac{h-2}{2} \right\rfloor + (h^2-3h+3) \left\lfloor \frac{h-1}{2} \right\rfloor \tag{3.1}$$

2. Van egy 2 hosszú zöld út, melynek belső csúcsához nem egyező útpár tartozik. Ehhez az útpárhoz maximum $\left\lfloor \frac{h}{2} \right\rfloor$ kódoló csúcs tartozik. Tetszőleges másik nem egyező párhoz legfeljebb $\left\lfloor \frac{h-1}{2} \right\rfloor$ kódoló csúcs lehet, itt az előző zöld utat kizárhatjuk. Az egyező párokhoz legfeljebb $\left\lfloor \frac{h-2}{2} \right\rfloor$ csúcs tartozhat. A kódoló csúcsok számát itt felülről becsüli
$$\left\lfloor \frac{h}{2} \right\rfloor + (h-2) \left\lfloor \frac{h-2}{2} \right\rfloor + (h^2-3h+2) \left\lfloor \frac{h-1}{2} \right\rfloor \tag{3.2}$$

A kódoló csúcsok száma $M(N) \leq \max\{(3.1),(3.2)\}$. Páratlan h-ra (3.1) nagyobb, mint (3.2), így:

$$M(N) \leq \binom{h-1}{2} + (h-3)\binom{h-3}{2} + (h^2-3h+3)\binom{h-1}{2} = (h^2-4h+5)\binom{h+1}{2}$$

Páros h-ra (3.2) nagyobb, mint (3.1), így:

$$M(N) \leq \binom{h}{2} + (h-2)\binom{h-2}{2} + (h^2-3h+2)\binom{h-2}{2} = (h^2-4h+5)\binom{h}{2}$$

Ezzel az állítást beláttuk. $\square$

4. fejezet

Algoritmusok a kódóló csúcsok maximális számának meghatározására

4.1. A keresés iránya

Az előző fejezetben megadtunk egy alsó és egy felső korlátot a kódoló csúcsok maximális számára. Az alsó h^2, a felső h^3 nagyságrendű. A pontos értékre jelenleg nem ismert általános formula. Az ismert pontos értékek $h = 2$ esetén 1, $h = 3$ esetén 4, $h = 4$ esetén 9, $h = 5$ esetén 16, végül $h = 6$ esetén 27 [2]. Ezek mind n^2 nagyságrendűek, $h \leq 5$ esetén értékük $(h-1)^2$.

Ebben a részben két algoritmust fogunk adni, melyek segítségével adott h esetén a kódoló csúcsok maximális száma megadható. Az algoritmusok kanonikus hálózatokat fognak generálni, majd ellenőrzik, hogy ezek minimálisak, így egyszerűek-e. Mindkét algoritmus programozásakor a C++ programozási nyelvet és a LEMON C++ template könyvtárat használtam.

Előbb azonban jöjjön egy tétel, mely a keresésben segít.

4.1.1. Tétel. *Legyen $N(G, s, T, h)$ egyszerű kódoló hálózat, melyben a kódoló csúcsok száma $C(N) > 0$. Ekkor létezik $N'(G', s, T, h)$ egyszerű hálózat, melyben a kódoló csúcsok száma $C(N) - 1$.*

Bizonyítás. Amint azt a 3.3.7 állításnál láttuk, van olyan v továbbító csúcs, melynek két kimenő éle a két célpontba vezet. Ennek bejövő éle a 3.3.5 állítás alapján

egy u kódoló csúcsból indul, és u két bejövő élén w_1 és w_2 továbbító csúcsok állnak. Készítsük N'-t N-ből a következő eljárással. Tegyük fel, hogy az u-n és v-n át t_1-be vezető, piros út a w_1 csúcsból, a t_2-be vezető, kék út pedig w_2-ből jön. Töröljük el az u és v csúcsokat, valamint éleiket, és a gráfhoz adjuk hozzá a (w_1, t_1) és a (w_2, t_2) éleket. Az így kapott N' hálózatban a kódoló csúcsok száma eggyel kevesebb, mint N-ben volt, és N' kanonikus hálózat, ahol létezik h éldiszjunkt út a célpontokba, amennyiben ez N-re is igaz volt. Ezek mellett N' egyszerű is, mivel N az volt, és az élek elhagyásánál nem keletkezhetett kör egyik segédgráfban sem, az új éleket pedig a célpontokba vezettük, melyekből nem vezet ki él, így itt sem lehet kör sem a hálózatban, sem a segédgráfokban. □

Ennek a tételnek az a haszna, hogy ha belátjuk, hogy nincs egy adott l számra olyan N minimális hálózat, melyben $C(N) = l$ kódoló csúcs van, akkor olyan sincs, melyben ennél több lenne. A keresésnél tehát, ha valamely l-re találtunk egy N minimális hálózatot, mely $C(N) = l$ kódoló csúccsal rendelkezik, valamint az algoritmussal beláttuk, hogy nincs $l + 1$ kódoló csúcsot tartalmazó h utas egyszerű hálózat, akkor ebből következik, hogy maximum l kódoló csúcs lehet egy ilyen hálózatban. Ezek alapján lehet a keresést a kódoló csúcsok száma szerint végezni, l-lel fölfelé haladva.

4.2. Az első algoritmus

Következzen az első algoritmus, mellyel a $\max(C(N))$ értéket akarjuk megtalálni valamely h esetén. Az algoritmus során veszünk egy rögzített l számot, ennyi kódoló csúcsa lesz a gráfoknak. Kezdetben adott s, t_1 és t_2 valamint h darab kezdőcsúcs az utaknak, ezekbe s-ből egy-egy él vezet. Jelöljük ezeket a csúcsokat $k_1, \ldots, k_h$-val. A 3.3.5 állítás szerint a k_i csúcsok továbbító csúcsok, kifokuk 2. A gráf ezután a 3.3.5 állítás alapján a következő egységekből épül fel. Minden egység két csúcsból fog állni, egy kódoló és egy továbbító csúcsból, a kódoló csúcsból él vezet a továbbítóba. A kódoló csúcs befoka és a továbbító csúcs kifoka 2, mivel kanonikus hálózatot építünk. Az egységeket ezentúl *pároknak*, a kódoló csúcsukat *alsó*, a továbbító csúcsukat *felső* csúcsnak nevezzük. A párok halmaza legyen $R = \{r_1, r_2, \ldots, r_l\}$. Minden pár pontosan egy t_1-be és pontosan egy t_2-be vezető úton kell, hogy rajta legyen, így készíthetünk két partíciót a párokból, ezek megmondják, hogy mely utakon vannak rajta. A gráf acklikus, így a csúcsoknak, de a pároknak is van topologikus sorrendje,

ezt fogja takarni a párok számozása.

Minden aciklikus hálózatot, mely elfogadott egy adott h-ra, és c kódoló csúcsot tartalmaz, megfeleltetünk egy struktúrának. Ez a struktúra a párok két, egymástól független partíciójából áll, mindkét partíció h osztállyal rendelkezik. A partíciók a célpontokba vezető utakat írják le. Az első partíció a t_1-be, a második a t_2-be vezető utakat képviseli, az osztályok reprezentálják azt, hogy az egyes utakon mely csúcspárok szerepelnek. Ezen osztályok közül mindkét partícióban legalább az egyik szükségképpen üres, a 3.3.6 állítás alapján. Minden partíciópárból legeneráljuk a neki megfelelő gráfot, majd ellenőrizzük, hogy ez minimális-e. A gráf építése úgy történik, hogy végigmegyünk a párokon, és ha egy pár az egyik partícióosztály első eleme, akkor az alsó csúcsába él vezet az osztálynak megfelelő út kezdőcsúcsából. Ha nem az első, akkor az osztály előző elemének felső csúcsából vezet ide él. Az osztály utolsó elemének felső csúcsából a megfelelő cépontba vezet él, üres út esetén pedig a kezdőcsúcsot kötjük össze a célponttal. A hálózat a konstrukcióból adódóan mindig elfogadott lesz.

A minimalitás ellenőrzéséhez végigiterálhatunk az éleken, egy lépésben egy élet elhagyva ellenőrizzük, hogy a hálózat elfogadott-e (az ellenőrzés után az élet vissza kell rakni). Ha volt olyan él, hogy azt elhagyva még mindig elfogadott a hálózat, akkor a hálózat nem minimális, egyébként igen. Az ellenőrzéshez használhatjuk a LEMON-ba épített, maximális folyam keresésére szolgáló algoritmust. Minden él kapacitását 1-nek választva, s kezdőponttal mindkét célpontba külön-külön futtatjuk az algoritmust, ha a folyam értéke mindkét esetben legalább h, a hálózat elfogadott. Az ellenőrzés történhet a $G_1(N)$ és $G_2(N)$ segédgráfok aciklikusságának ellenőrzésével is, ezt az eredeti gráfra nézzük, nem kell élet elhagyni.

4.3. Az első algoritmus implementációja

A megvalósításnál egy struktúrába tartozik a két partíció, ezek egészeket tartalamzó vektorok vektorai. Itt h darab, a pároknak megfelelő egész számokat tartalmazó vektor van, mindegyik tartalmazza a neki megfelelő partíciók elemeit. A partíciókon való iterálást megkönnyíti, ha nem rajtuk lépkedünk, hanem egy l elemű, egészeket tartalmazó vektoron, melynek minden eleme egy párnak felel meg, értéke azt jelöli, hogy a pár melyik partícióban van, így könnyebb sorba rendezni a felosztásokat. Az alább látható kódrészletben ezeknek az új vektoroknak a `helyek1` és `helyek2` vek-

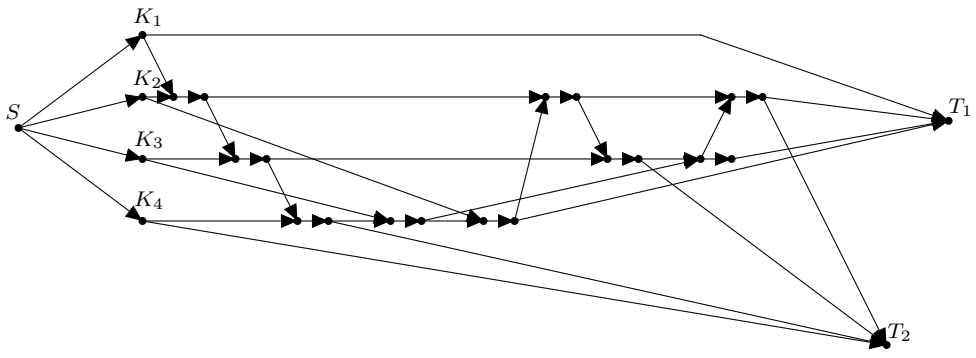
torok felelnek meg. A végigjárás két, egymásba ágyazott ciklussal történik, ezeknél az előbb említett vektorokon való lépéskor elsőként az első változóval végigfutunk h-ig, majd a másodikkal egyet lépünk, az elsővel megint h-ig futunk, és ezt addig ismételjük, míg végig nem érünk.

```
void general(int utakszama, int parokszama)
{
  vector<int> helyek1(parokszama+1,1);
  while(helyek1[parokszama]==1)
  {
    vector<int> helyek2(parokszama+1,0);
    while(helyek2[parokszama]==0)
    {
      //partíció készítése
      Particiok part(utakszama);
      for(int par=0; par<parokszama; par++)
      {
        part.elso[helyek1[par]].push_back(par);
        part.masodik[helyek2[par]].push_back(par);
      }
      Alapgraf g(utakszama,parokszama);
      if(epit(g,part,utakszama,parokszama))
      {
        if(minimal(g.graph,g.source,g.target1,g.target2,utakszama))
        {
          part.kiir();
        }
      }
      //következő partíciónak megfelelő vektorra átlépés
      int j=0;
      while(helyek2[j]==utakszama-2)
      {
        helyek2[j]=0;
        j++;
      }
      helyek2[j]++;
    }
    int i=0;
    while(helyek1[i]==utakszama-1)
    {
      helyek1[i]=1;
      i++;
    }
    helyek1[i]++;
  }
}
```

Az ellenőrzésnél a LEMON-ba beépített folyamalgoritmust célszerű használni [3].

Az algoritmus futása során, adott l esetén a vizsgált partíciók száma $(h-1)^{2l}$, emiatt a módszer csak kis h-k esetén hatékony. Például, $h = 4$-re az éles határ 9. Annak belátása, hogy $l = 10$-re nincs minimális gráf, mely 10 kódoló csúccsal rendelkezik és 4 éldiszjunkt út van mindkét célpontba, már 3^{20} eset.

A következő ábrán egy olyan hálózat gráfja látható, melyben $h = 4$, $l = 9$.



A következő algoritmus ugyan még ennél is több esetet néz végig, de nem kell minden lépésnél újra legenerálni az egész gráfot, mindig csak egy élet adunk hozzá a meglevőhöz, és akkor haladunk tovább, ha lehetséges.

4.4. A második algoritmus

Most kicsit máshogy épül fel a gráf. Kezdetben adott s, t_1 és t_2 valamint h darab kezdőcsúcs az utaknak, ezekbe s-ből egy-egy él vezet. Jelöljük ezeket a csúcsokat $k_1, \ldots, k_h$-val, mint az előző esetben. Itt is felosztjuk a párokat, azonban most csak egyetlen, h osztályból álló partíciót csinálunk, ahol az egyik osztály üres lesz, és az is marad. Most a csúcsok még nincsenek számozva, csak az fontos, melyik osztályban hány pár van. Elkészítünk egy vázat, mely a partíció alapján a következő: egy osztálynak megfelel egy (s, t_1) út, melyen annyi pár van, ahány elemű az osztály. Most beszámozzuk a csúcspárokat az első út első csúcsától kezdve folytatólagosan.

Ez után készítjük el a második partíciót, mely megmondja, hogy a t_2-be vezető utakon hány kódoló csúcs van. Végigmegyünk ezeken: vesszük az első utat, és úgy választjuk az első csúcsát, hogy az így kapott hálózat és közben az eredetivel együtt

megépített $G_1(N)$ és $G_2(N)$ segédgráfok aciklikusak maradjanak. Ezek közül azt választjuk, ahol minimális lesz az újonnan bevett csúcs sorszáma. Ezután megyünk a következő csúcsra, majd a következő útra, és így tovább. Ha elakadunk, de nem értünk végig, vagyis maradt csúcs amit nem használtunk fel, akkor az utolsónak betett, még jó csúcsot áthelyezzük a következő olyan helyre, ami megfelelő az aciklikusságok szempontjából.

4.5. A második algoritmus implementációja

Itt egy cilkuson belül történik a vizsgálat, a ciklus minden lépésében egy adott partícióját vesszük a csúcspároknak, ez a lenti kódrészletben az `felosztas` vektor. Ezután megépítjük a gráf vázát, a segédgráfokkal együtt. Ez a váz (s, t_1) utakat tartalmaz, melyeken a kezdőpontok és a csúcspárok szerepelnek. Feltehetjük most, hogy az utak sorba vannak rendezve a hosszuk szerint, az első kezdőcsúcshoz tartozzon a legrövidebb (ez biztosan nem tartalmaz kódoló csúcsot), a h-adikhoz pedig a leghosszabb. Ez után egy belső ciklus jön, itt létrehozzuk az l szám egy h elemű partícióját (egy lépésben egy ilyet vizsgálunk), mely megmondja, hogy a t_2-be vezető utak közül melyik kezdőcsúcshoz milyen hosszú út fog tartozni. Az alább látható kódrészletben ezt a `belsopart` vektor jelzi. Itt már nem tehetünk fel semmit a hosszukról. Az `Utak` struktura tartalmazza, hogy mely (s, t_2) utakon mely csúcsok szerepelnek, valamint ezen csúcsok sorrendjét is. A következő él beillesztéséhez végigiterálunk a csúcspárokon, és a még szabadon levők közül kiválasztjuk a legkisebb sorszámút, ezt a feladatot a `kovetkezo` függvény látja el, melynek visszatérési értéke `igaz`, ha sikerült, `hamis`, ha nem sikerült a művelet. Amennyiben nem sikerült új élen hosszabbítani az aktuálisan növelendő utat, akkor az út utolsó elemét a `leptet` függvény megpróbálja áthelyezni egy nagyobb sorszámú csúcspárra. Ha nem jár sikerrel, akkor ismét meghívódik, most még eggyel korábbi élre. Az ellenőrzés a segédgráfok aciklikusságának ellenőrzése, viszont ha eddig aciklikusak voltak, elég utat keresni az újonnan betett él vég- és kezdőpontja között.

```
void keres(int utakszama, int parokszama)
{
  const int alsolimit=1;
  vector<int> felosztas(utakszama,alsolimit);
  felosztas[0]=0;
  felosztas[utakszama-1]=parokszama-alsolimit*(utakszama-2);
  while(felosztas[0] == 0)
  {
    Alapgraf g(utakszama,parokszama);
    Residual1 res1(utakszama,parokszama);
    Residual2 res2(utakszama,parokszama);
    g.epit(felosztas);
    res1.epit(felosztas);
    res2.epit(felosztas);
    vector<int> belsopart(utakszama,alsolimit);
    belsopart[1]=0;
    belsopart[utakszama-1]=parokszama-alsolimit*(utakszama-2);
    bool vege=false;
    while(!vege)
    {
      Utak sorrend(parokszama);
      bool tovabb=true;
      while(tovabb)
      {
        if(sorrend.kovetkezo(g,res1,res2,belsopart,parokszama))
        {
          if(sorrend.perm.size() == parokszama)
          {
            befejez(g,res1,res2,belsopart);
            g.rajzol("graph.eps");
            res1.rajzol("residual1.eps");
            res2.rajzol("residual2.eps");
            tovabb=false;
            system("pause");
          }
        }
        else
        {
          bool b=false;
          while(!b)
          {
            b=sorrend.leptet(g,res1,res2,belsopart,parokszama);
            if(sorrend.perm.size()==0)
            {
              b=true;
              tovabb=false;
            }
          }
        }
      }
```

```
      vege=kovetkezo_masodikpart(belsopart, alsolimit);
    }
  kovetkezo_elsopart(felosztas);
  }
}
```

Az algoritmus még az előzőnél is több esetet jár végig, itt $l!$ nagyságrendű a vizsgált esetek száma. A futás során alkalmazott ellenőrzés azonban egyszerűbb, ha már foglalt a csúcs az élet máshová kell rakni, egyébként pedig egy szélességi bejárással, BFS-sel megoldható, ez a gyakorlatban gyorsabb a maximális folyamkereső algoritmusnál. Ezzel az algoritmussal is sikerült $h = 4$ esetén meghatározni a kódoló csúcsok maximális számát.

4.6. További célok

További kutatás irányai lehetnek hatékonyabb algoritmusok keresése, valamint a meglevők kiterjesztése a $k > 2$ esetre. Az alsó és felső korlátokra adott becslések javítása, valamint olyan esetek vizsgálata, ahol a gráf nem feltétlenül aciklikus. Olyan algoritmusok kifejlesztése, melyekkel $h = 5$ és $h = 6$, valamit további esetek is vizsgálhatók. Érdekes lehet a [2] szerzői által, a $h = 6$, $l = 27$ esetre megadott gráf konstrukciójának vizsgálata, általánosítása nagyobb gráfokra.

Irodalomjegyzék

[1] Michael Langberg, Alexander Sprintson, Jehoshua Bruck, *The Encoding Complexity of Network Coding*, IEEE Transactions on Information Theory, VOL. 52, NO. 6, JUNE 2006

[2] Li Xu, Weiping Shang, Guangyue Han, *A Graph Theoretical Approach to Network Encoding Complexity*, Information Theory and its Applications (ISITA), 2012 International Symposium on 28-31 Oct. 2012, Pages: 396 - 400

[3] LEMON, Library for Efficient Modeling and Optimization in Networks, EGRES, Egerváry Research Group on Combinatorial Optimization, 2003-2013, http://lemon.cs.elte.hu/