Szenzorhálózatok kombinatorikus problémái

Diplomamunka

Írta: Varga László

Alkalmazott matematikus szak

Témavezető:

Jordán Tibor, egyetemi docens
Operációkutatási Tanszék
Eötvös Loránd Tudományegyetem, Természettudományi Kar

Eötvös Loránd Tudományegyetem
Természettudományi Kar
2007

Tartalomjegyzék

1. fejezet

Szenzorhálózatok

A szenzorhálózatok számos érzékelő egységből felépített, önálló működésre képes elosztott számítógépes rendszerek. A hálózatot alkotó szenzorok a tér különböző pontjain helyezkednek el, megfigyeléseket végezhetnek, esetleg különféle tevékenységeket is végrehajthatnak. Legfontosabb feladatuk az adatgyűjtés, de emellett képesek lehetnek az adatokat feldolgozni és elemzéseket is végezni. Az érzékelő egységek általában olcsón előállíthatóak és aránylag kis méretűek, így a megfigyelt térrészben viszonylag sűrűn és nagy számban helyezkedhetnek el, ezzel lehetőség nyílik újfajta alkalmazások megvalósítására.

1.1. Felhasználási területek

A technológiai fejlődés előrehaladtával egyre szélesebb körben állítanak használatba vezeték nélküli szenzorhálózatokat, melyekben a szenzorok egymással együttműködve, valamilyen globális érzékelési feladat elvégzésére képesek.

- Számos hagyományos alkalmazásban van szükség térben elosztott módon mechanikai rezgéseket mérni, több száz vagy akár több ezer érzékelő elhelyezésével. Ilyenek például a szeizmográfiai mérések, műszaki felépítmények (pl. hidak) statikai és rezgésvizsgálatai, autó és repülőgép karosszériák tervezése során végzett rezgésvizsgálatok.

- Áruházak, raktárak, gyártósorok, kórházak, irodák, stb. logisztikai, információs rendszerei. Ezekben az alkalmazásokban előforduló szenzorok, beavatkozók és más elosztott csomópontok: vonalkód leolvasók, elektronikus árucímkék, hőmérséklet-érzékelők, léptető motorok, kézi számítógépek, irodai perifériák stb.

- Katonai alkalmazások: harcmezőn telepített (ad-hoc) mikrofonos szenzorhálózat a lövésirányok felderítésére, kooperatív járműrajok (tankhadoszlop, repülőkötelék) elosztott beágyazott rendszerekkel való támogatása.
- Kísérleti alkalmazások, pl. kooperatív robotok.
- Környezetvédelem: élőhely monitorozás, katasztrófa-előrejelzés.
- Precíziós mezőgazdasági termelés: hőmérséklet, légnyomás, nedvességtartalom, magasság mérés.

1.2. A szenzorhálózatok jellemzői

A szenzorokat gyakran nehezen megközelíthető helyekre telepítik, ezért az egységek pontos elhelyezése rendkívül nagy költségű, nagy körültekintést igénylő feladat. A szenzorhálózatok alkalmazása azonban még így is költséghatékonyabb megoldást jelent a hagyományos megfigyelési módszerekkel szemben.

Szükséges, hogy a szenzorok a környezeti hatásokkal szemben ellenállóak legyenek. Általában nincs lehetőség az egységek külön-külön történő javítására, cseréjére, ezért azoknak hosszú élettartamúaknak (több hónap vagy év) kell lenniük.

Ha elromlik egy szenzor, vagy lemerül a tápegysége, könnyen telepíthető a helyére egy újabb. Az egységeknek a rendszerbe való be-, illetve kilépését (valamint az akár nagyarányú bővítést) a hálózatok támogatják.

A szenzorok olcsón előállíthatóak, kis méretűek, egyéni azonosítókkal (ID) rendelkeznek, és az együttműködés során közös órajellel dolgoznak. A mérésekhez és a műveletek elvégzéséhez szükséges energiát elemek, ill. akkumulátorok biztosítják. A tápfeszültség szintjének szabályozásával biztosítható a szenzorok hosszú ideig tartó működése.

A hálózaton történő adattovábbítás jelentős energiabefektetést igényel, ezért szükséges a begyűjtött információ lokális előfeldolgoza, a redundanciák kiküszöbölése.

Léteznek vezetékes és vezeték nélküli hálózatok. Az előbbieknél az egységek közötti kapcsolat megteremtése költséges, körülményes feladat lehet, ráadásul a vezetékezés hibaforrássá is válhat (pl. rövidzár, szakadás). Az utóbbi megoldásnál általában rádiós összeköttetést alkalmaznak, ami jobban alkalmazkodik a környezeti feltételekhez, mint az infravörös vagy az ultrahangos jelű kommunikáció. A vezeték nélküli megoldás hátránya, hogy a

jelterjedést esetenként gátolhatják a környezeti akadályok, valamint a kommunikációs közeg.

Míg más hálózatokban jellemzően nagy hatósugarú rádióösszeköttetésre törekszenek, addig a szenzorhálózatokban többnyire elegendő a rövid hatósugár (max. 100 méter) használata. Ezzel is csökkenthető az energiafelhasználás, illetve kevésbé lépnek fel interferencia-problémák.

1.3. Felmerülő kérdések

A szenzorhálózatokkal kapcsolatban számos érdekes probléma fogalmazódott meg és vált vizsgálat tárgyává az elmúlt tíz-egynéhány év során. Elemzések tárgyát képezik többek között az alábbi témakörök:

- hálózatok telepítése, az egységek szinkronizálása
- szenzorok pozíciójának meghatározása (lokalizálás)
- kommunikációs rendszerek kiépítése, üzenetküldési protokollok meghatározása
- a hálózati topológia karbantartása
- adatgyűjtési stratégiák
- energiatakarékos üzemeltetés, élettartamnövelés
- mobil szenzorok alkalmazása
- adattömörítés, jelfeldolgozás, számítási eljárások
- hálózatvédelem

A publikált cikkek sokszor csak részeredményeket tartalmaznak, és a problémákat speciális esetekben tárgyalják, hiszen az alkalmazásokban ezek a megoldások is kiválóan használhatóak. A cikkekben tárgyalt, a gyakorlati életben felmerülő feladatok között viszont található számos, figyelemre méltó matematikai megfontolás, amit érdemes elemezni, pontosítani, a felmerülő kérdéseken elgondolkozni.

Az említett feladatok, témakörök közül hárommal fogunk részletesebben is foglalkozni, bemutatva egy-két kapcsolódó cikk ötleteit, algoritmusait – természetesen a leírtakat kissé átfogalmazva, érthetőbbé téve, a matematikai állításokat és gondolatmeneteket néhol pontosítva.

2. fejezet

Csatorna-beosztás, él-ütemezés

2.1. Bevezetés

A fejezet első részében a szenzorhálózatok (ill. tágabb értelemben a drótnélküli hálózatok) csatorna-beosztásáról lesz szó, mely témakör magába foglal több, a drótnélküli hálózatok témaköréhez tartozó problémát. Ezeket a feladatokat, sok más lehetséges problémával együtt, egy általános modell keretein belül mutatjuk be, és ismertetünk egy általános algoritmust, mely megoldást ad mindegyik feladatra. Az algoritmus, a gráfelméleti nyelven fogalmazva, pontszínezést, illetve élszínezést fog készíteni – csak éppen nem irányítatlan, hanem irányított gráfokra. A helyes színezés feltételei is bonyolultabbak lesznek, mint a hagyományos színezési feladatokban. Természetesen nem várhatunk optimális eredményt, lévén az általános feladat az NP-nehéz pontszínezés, illetve élszínezés feladatát speciális esetként tartalmazni fogja. Közelítő eredményeket fogunk kapni, különböző feladattípusokra másmás approximációt tudunk majd bizonyítani az algoritmus elemzésével.

A fejezet második részében kiválasztunk egy konkrét, szenzorhálózatokhoz kapcsolódó problémát az első részben említettek közül. Még ebben a speciális esetben is egy NP-beli problémával állunk szemben. Erre a feladatra adunk egy újabb algoritmust, mely nagyban felhasználja az irányítatlan gráfok élszínezésére vonatkozó ismeretünket, név szerint Vizing tételét [26], pontosabban egy, a tételre vonatkozó konstruktív bizonyítást. Vizsgálni fogjuk az algoritmus kimenetének viszonyát az optimális megoldáshoz.

2.2. Szenzorhálózatok csatorna-beosztása

Adott egy szenzorhálózat (vagy drótnélküli hálózat). Minden szenzornak adott a hatótávolsága. Ezzel adott az, hogy mely szenzorok tudják az általa küldött üzenetet fogadni.

Megjegyzés: Minden, amiről szó lesz, érvényes általában a *drótnélküli hálózatok*ra, de most speciálisan csak a szenzorhálózatok kontextusában vizsgálódunk.

2.2.1. A modell

$G = (V, A)$ irányított gráf, ahol V a szenzorok (pontok) halmaza, és (u, v) él, ha az u által küldött üzenetet v tudja fogadni. (Azaz u hatótávolsága akkora, hogy a küldött üzenet eljut v-hez.)

- Nem tesszük fel, hogy csak oda-vissza élek vannak. (Azaz nem tesszük fel, hogy minden szenzornak ugyanakkora a hatótávolsága.)
- Azt sem tesszük fel, hogy G erősen összefüggő, azaz hogy minden szenzor minden másiknak valamely útvonalon *elvileg* képes üzenetet eljuttatni. (Nem biztos, hogy erre igény van a hálózatban.)

A feladat ismertetése

Meg kell határozni a *lehető legkisebb* $q \in \mathbb{N}_+$-t, és ezzel egy q számú csatornából álló egységet a következőképpen: be kell osztani, hogy az egyes szenzorok vagy rendezett szenzorpárok (azaz irányított élek) az $\{1, \ldots, q\}$ csatornák közül melyet használhatják. Egy csatornát többen is használhatnak, és mindegyik szenzornak (vagy rendezett párnak) pontosan egyet kell használni. Lesznek bizonyos korlátozások arra, hogy mely szenzoroknak vagy mely szenzorpároknak tilos ugyanazt a csatornát használniuk.

Megjegyzés: Gondoljunk arra az esetre, amikor a szenzorok kommunikálni akarnak egymással. Ekkor a csatornák időegységeket jelentenek, a beosztás pedig azt mondja meg, hogy mely szenzorok mely időegységben küldhetnek-fogadhatnak üzenetet, illetve ha az éleket osztjuk be, akkor mely (irányított) éleken mely időegységben folyhat üzenetküldés. – Az iménti feladat egy (megengedett) megoldása nem ad protokollt arra, hogy ha el akarunk juttatni u-ból v-be egy üzenetet, azt mely útvonalon tegyük meg. Mindössze azt biztosítja, hogy ha adott egy ilyen protokoll, akkor (a q hosszú

időintervallumot egymás után többször használva) az üzenetküldés megvalósítható anélkül, hogy bizonyos üzenetküldési korlátozásokat megsértenénk (amivel például valamely szenzornál interferencia lépne fel, vagy más probléma adódna).

Sokféle feladatot meg lehet fogalmazni a csatorna-beosztás nyelvezetét használva. Egyszerűség kedvéért a továbbiakban gondoljunk a fenti példában említett szituációra, ahol a csatornák időegységeket jelképeznek, és a szenzorok üzeneteket küldenek egymásnak az egyes időegységekben.

Látjuk tehát, hogy kétféle feladat is lehet: vagy a pontokhoz, vagy az élekhez kell időegységet rendelni. Modellünkben ez azt jelenti, hogy meg kell adni egy pont- vagy élszínezést, csak sokkal általánosabb követelményeknek megfelelően, mint ahogy azt megszoktuk. A hagyományos színezési feladat irányítatlan gráfokra azt jelenti, hogy úgy kell megszínezni a pontokat/éleket, hogy a szomszédos pontok/élek különböző színűek legyenek. Most irányított gráfunk van, és nem csak a szomszédokra lehet korlátozásokat tenni. Lássuk tehát a korlátozásokat.

Korlátozások

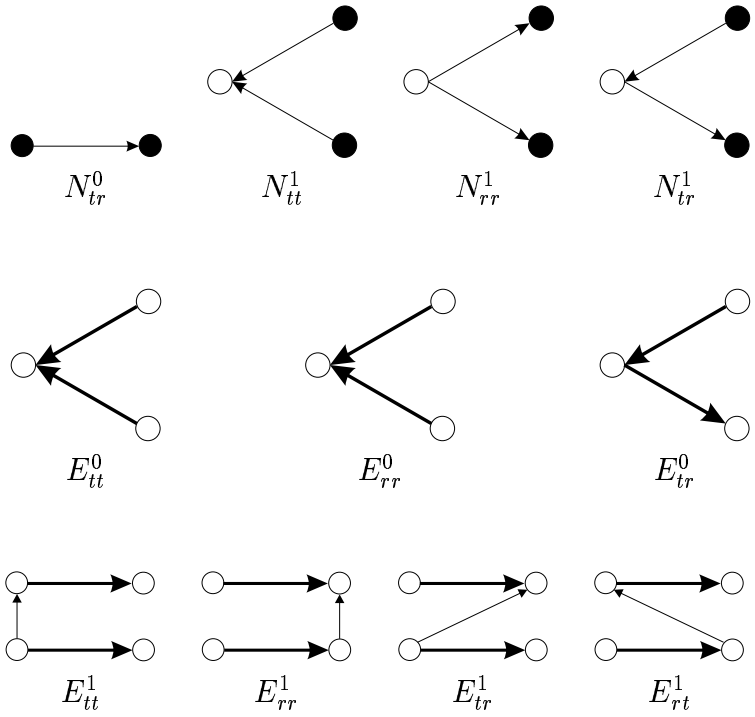
Jelentésük: ha *két pont/él korlátozásban áll*, akkor azokat különböző színnel kell színezni.

Korlátozás-sémákat adunk meg, és *két pont/él akkor áll* majd *korlátozásban egy c korlátozás(-séma) által*, ha c feltételei teljesülnek rájuk.

Egy c *korlátozás* a következőképp néz ki: $c = <\epsilon>^{<s>}_{<d>}$, ahol

- $\epsilon \in \{N, E\}$ jelzi, hogy pontpárra vagy élpárra vonatkozik a korlátozás
- $s \in \{0, 1\}$ jelzi a pontpár/élpár közötti kapcsolatot (azaz a 0 a pontok/élek szomszédosságát jelenti, az 1 pedig azt, hogy legyen köztük egy harmadik pont/él)
- $d \in \{tt, tr, rt, rr\}$ adja meg, hogy az s szerinti kapcsolatban mely irányba mutat(nak) az él(ek)

A korlátozások, melyekkel dolgozni fogunk, az alábbi ábrán láthatóak.



2.1. ábra. Korlátozások

Megjegyzés: *Érvénytelen korlátozások*: N^0_{tt} és N^0_{rr}, mert két szomszédos pont esetében a köztük menő élnek az egyik pont a kezdőpontja, a másik a végpontja kell, hogy legyen. (Ezek a korlátozások egy pontpárra sem teljesülnek, ezért nem kell foglalkozni velük.)

Megjegyzés: *Ekvivalens korlátozások*: egy pont-, ill. élpár pontosan akkor áll korlátozásban az egyik által, ha a másik által is. (Ezek közül egy feladatban elég az egyiket megadni.)

- N^0_{tr} és N^0_{rt}
- N^1_{tr} és N^1_{rt}
- E^0_{tr} és E^0_{rt}

Korlátozáshalmazokkal fogunk dolgozni. Egy *korlátozáshalmaz* a fenti sémák közül néhányat foglal magába.

Csak olyan korlátozáshalmazokkal foglalkozunk, melyekben vagy csak a pontokra, vagy csak az élekre vonatkozó korlátozások vannak. (A gyakorlati alkalmazások feladatai leírásában elegendő ilyeneket használni.)

Végül, *egy gráf pontjainak/éleinek színezése eleget tesz egy C korlátozáshalmaznak*, ha minden $c \in C$-re a c által korlátozott pontpárok/élpárok páronként különböző színűek. Ilyen színezést szeretnénk kapni.

2.2.2. A feladatunk pontosan

Adott egy $G = (V, A)$ irányított gráf (nincs benne hurokél és többszörös él), $\Psi \in \{N, E\}$ (pont- vagy élszínezésről van-e szó), és korlátozások egy C halmaza, ahol $C \subseteq \{\Psi^0_{tt}, \Psi^0_{rr}, \Psi^0_{tr}, \Psi^1_{tt}, \Psi^1_{rr}, \Psi^1_{tr}, \Psi^1_{rt}\}$.

Adjuk meg a pontok (ha $\Psi = V$) vagy az élek (ha $\Psi = E$) egy, a C-nek eleget tevő színezését, melyben a színek q száma minimális.

Jelölés: Legyen G_1 az a gráf, melyet G-ből úgy kapunk, hogy elhagyjuk az élek irányítását.

Megjegyzés: Tekintsük azt az esetet, amikor $C = \{E^0_{xy} : x, y \in \{t, r\}\}$. Ekkor a feladat azt jelenti, hogy G_1-ben keresünk egy (hagyományos értelemben vett) élszínezését a lehető legkevesebb színnel.

A $C = \{N^0_{xy} : x, y \in \{t, r\}\}$ esetben pedig arról van szó, hogy G_1-nek egy pontszínezését keressük, szintén minimális számú színnel.

A fenti két speciális eset NP-nehéz, ezért nem várhatunk az általános feladatra optimális megoldást biztosító, polinomiális időben futó eljárást.

Az alábbiakban az [22] cikkben szereplő polinomiális algoritmust mutatjuk be, melyre a különböző korlátozás-halmazok esetében különböző approximációs faktort tudunk majd biztosítani. Bizonyos tételekben konkrét felső becslést adunk a felhasznált színek maximális számára.

Megjegyzés: Az eljárás neve („UxDMA") onnan származik, hogy a csatornabeosztási feladatoknak a gyakorlatban három nagy típusa van, melyeket a TDMA, FDMA, illetve CDMA rövidítésekkel jelölünk, és az UxDMA algoritmus e három probléma közös általánosítására ad (megengedhető) megoldást. A rövidítések jelentése: a három probléma arról szól, hogy (minimális számú csatorna igénybevételével) közös csatorna-használatot akarunk engedélyezni a szenzorok számára (MA: Multiple Access), ahol csatorna alatt időt (Time), kódot (Code) vagy frekvenciát (Frequency) értünk.

2.2.3. Az UxDMA algoritmus

1. Rendezzük (rakjuk sorba) G pontjait.
2. Színezzünk (rendeljünk pozitív egész számokat a pontokhoz/élekhez), a rendezés szerint csökkenő sorrendben.

Pontszínezésnél az aktuális pontot színezzük, élszínezésnél a ponttal szomszédos (még színezetlen) éleket. A színezést mohón végezzük: az első színt választjuk (a legkisebb pozitív egészt), mellyel nem sérül egy korlátozás sem.

2.2.4. Az UxDMA elemzése

Az algoritmus kulcsa a rendezés lesz. Az [22]-ben háromféle rendezési szabályt említenek egy *irányítatlan gráf* pontjaira:

- **Random** (RAND): a pontok véletlen sorrendje.
- **Minimum Neighbors First** (MNF): a legkisebb fokú ponttól kezdve, a fokszámok szerinti növekvő sorrend.
- **Progressive Minimum Neighbors First** (PMNF): hasonló, mint az MNF, csak itt amikor egy pontnak sorszámot adtunk, utána a vele szomszédos éleket kidobjuk a gráfból, és az új fokszámok szerint vesszük a következő, legkisebb fokú pontot.

Az algoritmusban a rendezési szabályokat G-nek az irányítatlan G_1 változatára alkalmazhatjuk. A PMNF fog kiemelt szerepet kapni a későbbiekben. Ez a rendezési szabály a kombinatorikus optimalizálással foglalkozó írásokban általában **Minimum Degree Ordering** (MDO) néven szerepel, a továbbiakban mi is ezt az elnevezést használjuk.

Jelölések

ALG : az algoritmus által felhasznált színek száma egy adott feladatra,
OPT : ugyanezen feladat optimális megoldásában a színek száma.
Δ : a G-beli maximális fokszám (egy csúcs foka: befokának és kifokának összege).
ρ : a G-beli maximális befok
δ : a G-beli maximális kifok
$\delta_r := \max\{\frac{\rho}{\delta}, \frac{\delta}{\rho}\}$
Θ : G vastagsága, azaz a legkisebb t szám, hogy G felbontható t síkgráf unió jára.

Megjegyzés: $\delta_r \leq \min\{\rho, \delta\}$. Sőt, míg ρ és δ külön-külön nagy lehet, a köztük levő eltérés gyakran kicsi, tehát általában $\delta_r << \min\{\rho, \delta\}$.

Először olyan tételeket fogunk bizonyítani, melyek igazak lesznek, bármilyen rendezési szabályt használ is az algoritmus.

1. Tétel. *Tetszőleges rendezési szabállyal futtatva az UxDMA algoritmust,* $C \subseteq \{N^s_{xy}, x, y \in \{t, r\}, s \in \{0, 1\}\}$ *esetén* $ALG \leq \Delta^2 + 1$.

Bizonyítás: Legyen $x \in V$ tetszőleges. Felső becslést adunk azon y pontok számára, melyekhez létezik $c \in C$, hogy x és y korlátozásban áll c által. Az ennél eggyel nagyobb szám felső becslés lesz az x elem színére. Az x színét $color(x)$-szel jelöljük.

x-nek legfeljebb Δ szomszédja van, és legfeljebb $\Delta(\Delta - 1)$ másodszomszédja. Ezekkel állhat korlátozásban. Ez legfeljebb $\Delta + \Delta(\Delta - 1) = \Delta^2$ pontot jelent, vagyis $color(x) \leq \Delta^2 + 1$. Ez igaz minden $x \in V$-re, tehát $ALG \leq \Delta^2 + 1$. □

2. Tétel. *Tetszőleges rendezési szabállyal futtatva az UxDMA algoritmust,* $C \subseteq \{E^s_{xy}, x, y \in \{t, r\}, s \in \{0, 1\}\}$ *esetén* $ALG \leq \Delta^2 + (\Delta - 1)^2$.

Bizonyítás: Legyen $e \in A$ tetszőleges. Az előző bizonyításhoz hasonlóan, e-nek legfeljebb $2(\Delta - 1)$ él szomszédja, és legfeljebb $2(\Delta - 1)^2$ másodszomszédja. Ez legfeljebb $2(\Delta - 1) + 2(\Delta - 1)^2 = 2\Delta(\Delta - 1)$ élet jelent, vagyis $color(e) \leq 2\Delta(\Delta - 1) + 1 = \Delta^2 + (\Delta - 1)^2$. Ez igaz minden $e \in A$-ra, tehát $ALG \leq \Delta^2 + (\Delta - 1)^2$. □

3. Tétel. *Tetszőleges rendezési szabállyal futtatva az UxDMA algoritmust,* $\{N^1_{rr}, N^1_{tt}\} \not\subseteq C \subseteq \{N^0_{xy}, N^1_{xx} : x, y \in \{t, r\}\}$ *esetén* $\frac{ALG}{OPT} \leq \Delta + 1$.

Bizonyítás: *1. eset:* $N^1_{rr} \notin C$, $N^1_{tt} \notin C$

Hasonlóan az 1. Tétel bizonyításához, egy x pont legfeljebb Δ szomszédjával állhat korlátozásban, $color(x) \leq \Delta + 1$, ezzel $ALG \leq \Delta + 1$, $OPT \geq 1$, így $\frac{ALG}{OPT} \leq \Delta + 1$.

2. eset: $N^1_{rr} \in C$ vagy $N^1_{tt} \in C$

Tfh. egy x pont $v_1, \ldots, v_d$ szomszédai és $w_{ij} : 1 \leq i \leq d, 1 \leq j \leq m_i$ másodszomszédai azok, melyekkel x korlátozásban áll, és melyek már színezve vannak. Ekkor $color(x) \leq 1 + d + \sum_{i=1}^{d} m_i$.

Tfh. $N^1_{rr} \in C$ ($N^1_{tt} \in C$ hasonlóan megy). Azaz x a w_{ij}-kel az N^1_{rr} által korlátozásban áll. De ekkor egy konkrét i-re az $\{x\} \cup \{w_{ij} : 1 \leq j \leq m_i\}$

halmaz pontjai páronként korlátozásban állnak egymással az N^1_{rr} által, mert mindből megy él v_i-be. Tehát ezeknek különböző színe kell legyen minden, a C-nek eleget tevő színezésben. Ez igaz minden i-re, vagyis $OPT \geq \max\{m_1+1, \ldots, m_d+1\} = 1+\max\{m_1, \ldots, m_d\}$.

Ezzel $color(x) \leq 1+d+\sum_{i=1}^{d} m_i \leq 1+d\cdot(1+\max\{m_1,\ldots,m_d\}) \leq 1+d\cdot OPT \leq 1+\Delta\cdot OPT$. Azaz $ALG \leq 1+\Delta\cdot OPT$, vagyis $\frac{ALG}{OPT} \leq \Delta+1$. □

4. Tétel. *Tetszőleges rendezési szabállyal futtatva az UxDMA algoritmust,* $C \subseteq \{N^s_{xy} : s \in \{0,1\}, x,y \in \{t,r\}\}$ *esetén, ha* G*-ben csak oda-vissza élek vannak, akkor* $\frac{ALG}{OPT} \leq \Delta+1$.

Bizonyítás: Mivel G-ben csak oda-vissza élek vannak, ezért a $\{N^s_{xy} : s \in \{0,1\}, x,y \in \{t,r\}\}$ halmaz bármely két eleme ekvivalens. Így az előző bizonyításhoz csak azt kell hozzátenni, hogy a *2. eset*-ben ha valamely 1-es típusú korlátozás benne van C-ben, akkor feltehető, hogy az összes többi is benne van – így megmarad annak a lépésnek a helyessége, hogy „egy konkrét i-re az $\{x\} \cup \{w_{ij} : 1 \leq j \leq m_i\}$ halmaz pontjai páronként korlátozásban állnak egymással az N^1_{rr} által". □

5. Tétel. *Tetszőleges rendezési szabállyal futtatva az UxDMA algoritmust,* $\{E^0_{xy} : x,y \in \{t,r\}\} \subseteq C \subseteq \{E^s_{xy} : s \in \{0,1\}, x,y \in \{t,r\}\}$ *esetén* $\frac{ALG}{OPT} \leq 2\Delta-1$.

Bizonyítás: A 2. Tétel miatt $ALG \leq \Delta^2+(\Delta-1)^2$. $\{E^0_{xy} : x,y \in \{t,r\}\} \subseteq C$ miatt $OPT \geq \Delta$.

Tehát $\frac{ALG}{OPT} \leq \frac{\Delta^2+(\Delta-1)^2}{\Delta} = \Delta + \frac{\Delta^2-2\Delta+1}{\Delta} = \Delta+\Delta-2+\frac{1}{\Delta} \leq 2\Delta-1$. □

6. Tétel. *Tetszőleges rendezési szabállyal futtatva az UxDMA algoritmust,* $C = \{E^0_{xy} : x,y \in \{t,r\}\}$ *esetén* $\Delta \leq ALG \leq 2\Delta-1$.

Bizonyítás: Egy él legfeljebb a $2(\Delta-1)$ szomszédjával állhat korlátozásban, tehát $ALG \leq 2\Delta-1$, és $ALG \geq OPT \geq \Delta$. □

Megjegyzés: A 6. Tétel feladata G_1 (hagyományos értelemben vett) élszínezésével ekvivalens. Erre a feladatra a 2.3-as szakaszban látni fogunk egy (szintén polinomiális) algoritmust, melyben a felhasznált színek száma csak $\Delta+1$ lesz.

Most rátérünk azokra a tételekre, melyekben a MDO rendezési szabályt használva, jobb becsléseket tudunk adni. Fel fogjuk használni a Θ paramétert, mely a gráf síkbeliségének „mértékét" jelzi.

Megjegyzés: A Θ kiszámítása NP-nehéz feladat, de erre az UxDMA algoritmusnak nincs is szüksége. Valójában egy másik paramétert is felhasználhatnánk, de erre csak a tételek bizonyítása után térünk ki.

Szükségünk lesz két lemmára, melyek a Θ-ról szólnak.

1. Lemma. *Minden irányítatlan $G(V,E)$ gráfban, melynek Θ a vastagsága, van olyan pont, melynek foka legfeljebb* $6\Theta - 1$.

Bizonyítás: G vastagsága Θ, azaz $G = G_1 \cup \ldots \cup G_\Theta$, ahol minden $G_i = (V(G_i), E(G_i))$ síkgráf. Ismert: $E(G_i) \leq 3V(G_i) - 6$. Ezt minden i-re összegezve $\sum_{i=1}^{\Theta} E(G_i) \leq \sum_{i=1}^{\Theta}(3V(G_i) - 6) \Rightarrow E \leq \sum_{i=1}^{\Theta}(3V - 6) = (3V - 6)\Theta$. Ha G-ben minden pont foka legalább 6Θ volna, akkor $2E \geq 6\Theta V$ volna, ellentmondás. □

2. Lemma. *Egy irányítatlan $G(V,E)$ gráf pontjainak MDO szerinti rendezése után minden pontnak legfeljebb* $6\Theta - 1$ *olyan szomszédja lesz, melynek nála nagyobb sorszáma van.*

Bizonyítás: Tekintsük a j-edik sorszámot kapott x pontot. Legyen $G^{(j)}$ az a gráf, melyet G-ből az első $j-1$ pont törlésével kaptunk. Ekkor $G^{(j)} \subseteq G$, ezért $G^{(j)}$ vastagsága legfeljebb Θ. A rendezési szabály szerint x minimális fokú $G^{(j)}$-ben, ezért az 1. Lemma alapján $G^{(j)}$-ben x foka legfeljebb $6\Theta - 1$. Vagyis a nála nagyobb sorszámú szomszédai legfeljebb ennyien vannak. □

7. Tétel. *A MDO rendezési szabállyal futtatva az UxDMA algoritmust, $C \subseteq \{N_{xy}^0 : x, y \in \{t, r\}\}$ esetén $ALG \leq 6\Theta - 1$.*

Bizonyítás: Amikor egy x pontot színezünk, akkor csak azon, vele korlátozásban álló szomszédjainak színét nem adhatjuk neki, melyeknek nála nagyobb sorszámúak. (A többi szomszédjának még nincs színe.) Ezek a 2. Lemma alapján legfeljebb $6\Theta - 1$-en vannak. Ez felső becslés $color(x)$-re, és egyúttal az ALG-ra. □

8. Tétel. *A MDO rendezési szabállyal futtatva az UxDMA algoritmust, $C \subseteq \{N_{xy}^s : s \in \{0,1\}, x, y \in \{t, r\}\}$ esetén, ha N_{rr}^1 vagy N_{tt}^1 benne van C-ben, akkor $\frac{ALG}{OPT} \leq 12\Theta(1 + \delta_r)$.*

Bizonyítás: Tekintsük azt, amikor egy bizonyos x pontot színezünk az algoritmusban. x szomszédai közül $\{p_1, \ldots, p_k\}$ nála kisebb sorszámú, $\{q_1, \ldots, q_n\}$ nála nagyobb sorszámú. Még csak a q_j-knek van színe. Jelölje $N(p_i)$ és $N(q_j)$ rendre a p_i és q_j azon szomszédainak halmazát, melyek x-nek *nem* szomszédai, már van színük, és x-szel (nyilván 1-es típusú) korlátozásban állnak. Ezzel $color(x) \leq 1+n+\sum_{i=1}^{k} |N(p_i)|+\sum_{j=1}^{n} |N(q_j)| \leq 1+n+k\cdot\max_i(|N(p_i)|)+ n \cdot \max_j(|N(q_j)|)$.

$n \leq 6\Theta - 1$ a 2. Lemma miatt, $\max_j(|N(q_j)|) \leq \Delta - 1$ és $k \leq \Delta$ nyilvánvaló.

Állítás: $\max_i(|N(p_i)|) \leq 6\Theta - 2$.

Bizonyítás: Tekintsünk egy p_i-t és ennek egy $y \in N(p_i)$ szomszédját. y már színezett, ezért sorszáma nagyobb, mint x-é (ezt jelölje $y > x$), de $x > p_i$, ezzel $y > p_i$. p_i-nek a 2. Lemma szerint legfeljebb $6\Theta - 1$ szomszédja van, mely nála nagyobb sorszámú, ezek közül x ilyen, tehát $|N(p_i)| \leq 6\Theta - 2$. Ez igaz minden i-re. $\square_{\text{Állítás}}$

A fenti becsléseket alkalmazva kapjuk: $color(x) \leq 1 + 6\Theta - 1 + \Delta(\Theta - 2) + (6\Theta - 1)(\Delta - 1) = 6\Theta + 12\Theta\Delta - 3\Delta - 6\Theta + 1 \leq 12\Theta\Delta$.

N^1_{rr} vagy N^1_{tt} benne van C-ben, ezért $OPT \geq \min(\rho, \delta)$, mivel a ρ-t adó élek (páronként) az N^1_{tt}, a δ-t adó élek (páronként) az N^1_{rr} által korlátozásban állnak egymással. Mivel $\Delta \leq \rho + \delta$, ezért $color(x) \leq 12\Theta(\rho + \delta)$, ami persze ALG-ra is igaz.

Így tehát $\frac{ALG}{OPT} \leq \frac{12\Theta(\rho+\delta)}{\min\{\rho,\delta\}} = 12\Theta \cdot \max\{1 + \frac{\delta}{\rho}, 1 + \frac{\rho}{\delta}\} = 12\Theta(1 + \delta_r)$. $\square$

9. Tétel. *A MDO rendezési szabállyal futtatva az UxDMA algoritmust,* $C \subseteq \{E^s_{xy} : s \in \{0,1\}, x, y \in \{t, r\}\}$ *esetén* $ALG \leq 36\Theta\Delta$.

Bizonyítás: Tekintsük azt, amikor egy $e = x\vec{y}$ élet színezünk. Legyen például $y > x$, vagyis az e élet akkor színezzük, amikor az MDO szerinti sorrendben az y következik.

Az e-vel korlátozásban álló élek közül (az esetleges $y\vec{x}$-et nem számítva) nevezzük *x-éleknek* az x végpontúakat és még azokat, melyeket x-től egy másik él elválaszt. Az *y-éleket* hasonlóképp definiáljuk. E két halmaz méretére adunk felső becslést.

Nézzük az y-éleket. (Az x-élekre ugyanez a gondolatmenet és becslés alkalmazható.)

Legyenek $\{p_1, \ldots, p_k\}$ y szomszédai közül azok, melyeknek kisebb a sorszámuk y-nál, és $\{q_1, \ldots, q_n\}$ az y-nál nagyobb sorszámúak. A q_j-knél lévő éleket már mind megszíneztük, ezek számát (azaz a q_j fokát) jelöljük $d(q_j)$-vel. Esetleg már néhány élet a p_i-knél is megszíneztünk, ezek halmazát jelöljük $I(p_i)$-vel. Ezzel az y-élek száma $\leq \sum_{j=1}^{n} d(q_j) + \sum_{i=1}^{k} |I(p_i)|$.

Itt $n \leq 6\Theta - 1$ a 2. Lemma miatt, $d(q_j) \leq \Delta$ és $k \leq \Delta$ pedig nyilvánvaló.
Állítás: $|I(p_i)| \leq 2(6\Theta - 1)$.
Bizonyítás: Tekintsünk egy színezett $p_i z$ élt ($z = y$ is lehet, és bármelyik irányba mutathat az él). Mivel ez színezett, ezért vagy $z = y$ és ekkor $z > p_i$ (a sorszámukra értve), vagy a $p_i z$ él valamelyik végpontjának sorszáma nagyobb kell legyen y-énál, ez pedig csak z lehet – és így is $z > p_i$. Ilyen z szomszédja p_i-nek legfeljebb $6\Theta - 1$ lehet, és ilyen $p_i z$ él legfeljebb $2(6\Theta - 1)$ lehet. $\square_{\text{Állítás}}$

A fenti becsléseket beírva, az y-élek száma $\leq \Delta \cdot (6\Theta - 1) + 2 \cdot \Delta \cdot (6\Theta - 1) = 3\Delta \cdot (6\Theta - 1)$.

Figyelembe véve az $y\vec{x}$ élet is, $color(e) \leq 2 \cdot 3\Delta \cdot (6\Theta - 1) + 1 + 1 \leq 36\Theta\Delta$. A tételt beláttuk. $\square$

10. Tétel. *A MDO rendezési szabállyal futtatva az UxDMA algoritmust,* $\{E^0_{xy} : x, y \in \{t, r\}\} \subseteq C \subseteq \{E^s_{xy} : s \in \{0, 1\}, x, y \in \{t, r\}\}$ *esetén* $\frac{ALG}{OPT} \leq 36\Theta$.

Bizonyítás: A 9. Tétel miatt $ALG \leq 36\Theta\Delta$. A feltétel miatt $OPT \geq \Delta$, ezzel kész. $\square$

Megjegyzés: Amint láthattuk, a Θ paraméternek csak azt a tulajdonságát használtuk ki a fenti bizonyításokban, hogy a $6\Theta - 1$ szám felső korlát az MDO rendezésben egy pont nagyobb sorszámú szomszédainak számára. Egy konkrét gráfra viszont polinom időben kiszámolható az MDO sorrend, és az imént említett szomszédságok maximuma is. (Bár egy gráfnak esetleg lehet többféle MDO sorrendje is, melyekben ez a maximum eltérő lehet.) Ezt a paramétert mondjuk p-vel jelölve, az UxDMA algoritmusra a fenti tételekben Θ helyett $\frac{p+1}{6}$-ot írva jobb becsléseket kapunk.

2.2.5. A Θ paraméter

A fentiekben felhasználtuk a Θ paramétert, ezért érdemes összefoglalva áttekinteni a vele kapcsolatos fontosabb ismereteket. Az alábbi eredményeket a [1],[4],[6],[7],[8],[16] cikkek alapján közöljük. A $G = (V, E)$ irányítatlan gráfban legyen $n = |V|, e = |E|$, Δ továbbra is a G-beli maximális fokszám, valamint $\Theta(G)$ a G gráf vastagsága. K_n-nel a teljes n pontú gráfot, $K_{m,n}$-nel az m és n elemszámú ponthalmazokon vett teljes páros gráfot jelöljük.

- $\Theta(G) = \max\{\Theta(H) : H \text{ a } G\text{-nek egy 2-összefüggő komponense}\}$
- $H \subseteq G \Rightarrow \Theta(H) \leq \Theta(G)$
- Az Euler-tétel segítségével igazolható:
  - Minden G gráfra $\Theta(G) \geq \left\lceil \frac{e}{3n-6} \right\rceil$.
  - Háromszögmentes G-re $\Theta(G) \geq \left\lceil \frac{e}{2n-4} \right\rceil$.
- $\Theta(K_9) = \Theta(K_{10}) = 3$, és ha $n \neq 9, 10$, akkor $\Theta(K_n) = \left\lfloor \frac{n+2}{6} \right\rfloor$.
- $\Theta(K_{m,n}) = \left\lceil \frac{m \cdot n}{2(m+n-2)} \right\rceil$, kivéve ha $m \leq n$ páratlanok és létezik $k \in \mathbb{N}$: $n = \left\lfloor \frac{2k(m-2)}{m-2k} \right\rfloor$.
- $\Theta(G) \leq \left\lfloor \frac{n+2}{6} \right\rfloor$ (mert $\Theta(G) \leq \Theta(K_n)$)
- $\Theta(G) \leq \left\lfloor \sqrt{\frac{e}{3}} + \frac{3}{2} \right\rfloor$
- $\Theta(G) \leq \frac{\sqrt{e}}{16} + O(1)$
- $\Theta(G) \leq \left\lceil \frac{\Delta}{2} \right\rceil$
- G fa-vastagsága $\leq k \Rightarrow \Theta(G) \leq \left\lceil \frac{k}{2} \right\rceil$
- G-ben nincs K_5 minor $\Rightarrow \Theta(G) \leq 2$.

A [22] cikkben azt állítják, egy gráf vastagsága általában sokkal kisebb, mint a benne levő maximális fokszám. A fenti eredmények ezt az állítást nem támasztják alá. Azonban, a [22]-ben kis gráfokon demonstrálják (a gráf vastagságára felső becslést adva), hogy a Θ általában valóban jóval kisebb, mint a Δ. Állítják továbbá, hogy a gyakorlati feladatokban a Θ egy kis konstanssal becsülhető.

Összefoglalva, a következő tételeket bizonyítottuk:

	Korlátozások (C) és Feltétel	Felhasznált rendezési szabály	Eredmény
1.	$C \subseteq \{N^s_{xy}, x, y \in \{t,r\}, s \in \{0,1\}\}$	tetszőleges	$ALG \leq \Delta^2 + 1$
2.	$C \subseteq \{E^s_{xy}, x, y \in \{t,r\}, s \in \{0,1\}\}$	tetszőleges	$ALG \leq \Delta^2 + (\Delta-1)^2$
3.	$\{N^1_{rr}, N^1_{tt}\} \not\subseteq C \subseteq \{N^0_{xy}, N^1_{xx} : x, y \in \{t,r\}\}$	tetszőleges	$\frac{ALG}{OPT} \leq \Delta + 1$
4.	$C \subseteq \{N^s_{xy} : s \in \{0,1\}, x, y \in \{t,r\}\}$ Feltétel: G-ben csak oda-vissza élek vannak	tetszőleges	$\frac{ALG}{OPT} \leq \Delta + 1$
5.	$\{E^0_{xy} : x, y \in \{t,r\}\} \subseteq C \subseteq \{E^s_{xy} : s \in \{0,1\}, x, y \in \{t,r\}\}$	tetszőleges	$\frac{ALG}{OPT} \leq 2\Delta - 1$
6.	$C = \{E^0_{xy} : x, y \in \{t,r\}\}$	tetszőleges	$\Delta \leq ALG \leq 2\Delta - 1$
7.	$C \subseteq \{N^0_{xy} : x, y \in \{t,r\}\}$	MDO	$ALG \leq 6\Theta - 1$
8.	$C \subseteq \{N^s_{xy} : s \in \{0,1\}, x, y \in \{t,r\}\}$ Feltétel N^1_{rr} vagy N^1_{tt} benne van C-ben	MDO	$\frac{ALG}{OPT} \leq 12\Theta(1+\delta_r)$
9.	$C \subseteq \{E^s_{xy} : s \in \{0,1\}, x, y \in \{t,r\}\}$	MDO	$ALG \leq 36\Theta\Delta$
10.	$\{E^0_{xy} : x, y \in \{t,r\}\} \subseteq C \subseteq \{E^s_{xy} : s \in \{0,1\}, x, y \in \{t,r\}\}$	MDO	$\frac{ALG}{OPT} \leq 36\Theta$

Ezen szakasz zárásaként, tekintsünk nyolc valós életbeli csatorna-beosztási feladatot. A hagyományos feladat-leírásokat mellőzzük, inkább a tárgyalt modellünk keretein belül írjuk fel őket, a megfelelő korlátozáshalmazokkal:

1.	N^0_{tr}	Cellular network Freq. assignment
2.	N^1_{tt}	TOCA/ROCA CDMA code assignment
3.	N^0_{tr}, N^1_{tt}	(T/F)DMA broadcast schedule/assignment
4.	$E^0_{rr}, E^0_{tt}, E^0_{tr}$	POCA CDMA code assignment
5.	$E^0_{rr}, E^0_{tt}, E^0_{tr}, E^1_{tr}$	(T/F)DMA link schedule/assignment
6.	$E^0_{rr}, E^0_{tt}, E^1_{tr}$	Full Duplex (T/F)DMA link schedule/assignment
7.	E^0_{rr}, E^0_{tr}	(T/F)DMA schedule/assignment with multiple directed antennas
8.	$E^0_{rr}, E^0_{tt}, E^0_{tr}, E^1_{tt}$	RTS-CTS protocols

A bizonyított tételekből ezekre a feladatokra a következő eredmények adódnak:

Feladat sorszáma	Korlátozás-halmaz	Tetszőleges rendezés-sel	MDO-t használva
1.	N_{tr}^0	$\frac{ALG}{OPT} \leq \Delta + 1$	$ALG \leq 6\Theta - 1$
2.	N_{tt}^1	$\frac{ALG}{OPT} \leq \Delta + 1$	$\frac{ALG}{OPT} \leq 12\Theta(1+\delta_r)$
3.	N_{tr}^0, N_{tt}^1	$\frac{ALG}{OPT} \leq \Delta + 1$	$\frac{ALG}{OPT} \leq 12\Theta(1+\delta_r)$
4.	$E_{rr}^0, E_{tt}^0, E_{tr}^0$	$ALG \leq 2\Delta - 1$	$ALG \leq 2\Delta - 1$
5.	$E_{rr}^0, E_{tt}^0, E_{tr}^0, E_{tr}^1$	$\frac{ALG}{OPT} \leq 2\Delta - 1$	$\frac{ALG}{OPT} \leq 36\Theta$
6.	$E_{rr}^0, E_{tt}^0, E_{tr}^1$	$ALG \leq \Delta^2 + (\Delta - 1)^2$	$ALG \leq 36\Theta\Delta$
7.	E_{rr}^0, E_{tr}^0	$ALG \leq \Delta^2 + (\Delta - 1)^2$	$ALG \leq 36\Theta\Delta$
8.	$E_{rr}^0, E_{tt}^0, E_{tr}^0, E_{tt}^1$	$\frac{ALG}{OPT} \leq 2\Delta - 1$	$\frac{ALG}{OPT} \leq 36\Theta$

2.3. Él-ütemezés szenzorhálózatokban

Az alábbiakban a fejezet előző részének végén említett feladatok közül az 5. sorszámúval fogunk részletesebben foglalkozni.

Adott egy szenzorhálózat, rajta egy TDMA (Time Division Multiple Access) protokoll. Ez azt jelenti, hogy (általunk meghatározható mennyiségű időegységből álló) szakaszokra oszthatjuk az időt, és megmondhatjuk, hogy az egyes szakaszokban mely szenzorok között mely időegység(ek)ben történhet üzenetküldés.

A hálózatot továbbra is a fejezet előző részében definiált $G = (V, A)$ irányított gráffal modellezzük.

Él-ütemezés: időegységek hozzárendelése a hálózatban a szomszédos pontokhoz (vagyis a G-beli irányított élekhez). Meg kell határozni az időszakaszok hosszát is, amellyel dolgozunk.

2.3.1. A feladat

Az él-ütemezés nyelvezetén megfogalmazva: Adott szenzorhálózathoz készítsünk él-ütemezést, a lehető legkevesebb időegységből álló idő-szakaszt használva. Ezzel azt határozzuk meg minden párra, hogy az első tagja a második tagjának mely időegységben küldhet üzenetet. Vigyáznunk kell, hogy ha a hálózatban az üzenetküldések elindulnak, ne fordulhasson elő egy szenzornál sem interferencia. Ezért nem rendelhetjük ugyanazt az időegységet két párhoz, ha valamelyik szenzor mindkét párban szerepel, vagy ha az egyik pár első tagja képes üzenetet küldeni a másik pár második tagjának.

A csatorna-beosztási modell nyelvezetén megfogalmazva: Adott egy G irányított gráf, színezzük meg minél kevesebb színnel az éleit, a $C = \{E^0_{rr}, E^0_{tt}, E^0_{tr}, E^1_{tr}\}$ korlátozáshalmaznak eleget tevően (azaz a szomszédos élek különböző színűek legyenek, és két él ne legyen egyszínű, ha az egyik kezdőpontjából a másik él végpontjába mutat egy él).

Feladatunk tehát egy speciális csatorna-beosztási feladat, de még így is NP-nehéz [22]. A fejezet előző részében bemutatott algoritmus természetesen alkalmazható a feladat megoldására, és azt is bizonyítani tudtuk, hogy tetszőleges rendezéssel futtatva $(2\Delta - 1)$-közelítő, MDO-val futtatva (36Θ)-közelítő.

Most egy másik algoritmust ismertetünk, a [10] alapján. A bemutatandó eljárás kétfázisú, és elosztottan működik (bár a második fázis leírását csak centralizáltan adjuk meg).

Jelölje $G_2(V, A_2)$ azt a gráfot, melyet a korábbiakban definiált G_1-ből úgy kapunk, hogy a párhuzamos éleket egy-egy éllel helyettesítjük, és legyen Δ_2 a G_2-beli maximális fokszám. (Emlékeztető: G_1 az a gráf, melyet G-ből úgy kapunk, hogy elhagyjuk az élek irányítását.)

Az algoritmus az első fázisban egy (hagyományos értelemben vett) élszínezést csinál G_2-n, legfeljebb $\Delta_2 + 1$ színnel. Ez a fázis Misra és Gries algoritmusán alapul. A második fázisban készül el az él-ütemezés.

Bizonyítani fogjuk, hogy ha G_2 egy fa, akkor az algoritmus legfeljebb $\Delta+2$ időegységből álló idő-szakaszt készít, ami lényegesen jobb, mint az UxDMA algoritmus eredménye. Δ természetesen egy alsó korlát az időegységek számára, hiszen a szomszédos éleknek különböző színűeknek kell lenniük.

Megjegyzés: Ha G_2 nem fa, akkor nem tudunk semmilyen biztosítékot mondani arra vonatkozólag, hogy hány időegységből kell állnia az él-ütemezés idő-szakaszának. A [10]-ben aránylag kevés élt tartalmazó (ritka) nem-fa gráfokra szimulációval azt a feltevést támaszották alá, hogy az algoritmus által felhasznált időegységek száma közel van $\Delta + 2$-höz. Azt mondhatjuk tehát, hogy nem csak a fáknál, de a gyakorlati feladatokban előforduló gráfoknál is jól működik az algoritmus.

2.3.2. Konstruktív bizonyítás Vizing tételére

Feladat: Adott egy egyszerű irányítatlan gráf. Színezzük meg az éleit legfeljebb $D + 1$ színnel, ahol D a gráfban a maximális fokszám.

Az alábbiakban Misra és Gries algoritmusát [19] mutatjuk be a feladat megoldására. Ezen eljárás elosztott változatát fogjuk majd alkalmazni az eredeti feladatunkban az él-ütemezéshez készítendő élszínezésnél.

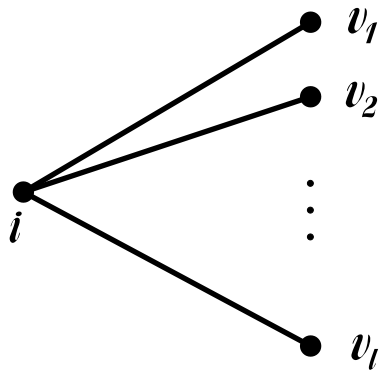
Az algoritmus egyenként színezi meg az éleket, végig fenntartva a *színezés helyességét*, azaz a már színezett élek között nincs két szomszédos, melyek azonos színűek.

Két adatstruktúrára lesz szükség: a legyezőre és a *cd*-útra.

Egy i pont $< v_1 \dots v_l >$ *legyezője* (2.2. ábra) az i néhány szomszédjának egy olyan listája, mely

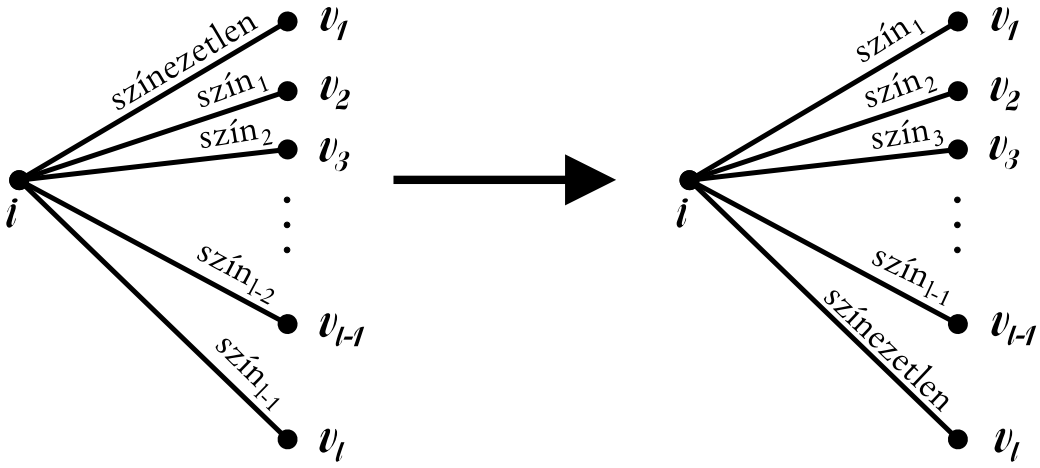
- nemüres
- az (i, v_1) él nincs színezve
- az (i, v_{j+1}) él színe szabad a v_j pontnál $(j = 1, \dots, l-1)$.

Bármely ponthoz egy maximális legyezőt mohó módon meg lehet konstruálni.



2.2. ábra. Legyező

A $< v_1 \dots v_l >$ *legyező forgatása* alatt azt értjük, hogy minden j-re az (i, v_j) élet átszínezzük az (i, v_{j+1}) él színére (2.3. ábra). Ekkor a színezetlen (i, v_1) él megszíneződik, az (i, v_l) él elveszti a színét. Könnyen belátható, hogy művelet megőrzi a gráf színezésének helyességét.



2.3. ábra. Legyező forgatása

Ha adott egy i pont, és nála egy $< v_1 \dots v_l >$ legyező, akkor *cd-út* alatt egy olyan, i-ből induló, c és d színű élekből álló, maximális hosszú (alternáló) utat értünk, ahol c szabad szín i-nél, d pedig v_l-nél. (A $c = d$ eset is megengedett, ekkor persze a cd-út hossza 0.) Ez egyértelműen létezik (adott

$i, < v_1 \dots v_l >, c, d$ és egy helyes színezés esetén). Az *út invertálása* alatt azt értjük, hogy a c színű éleket d színűekre cseréljük rajta és fordítva. Egyszerűen igazolható, hogy ez a művelet megőrzi a gráf színezésének helyességét.

Legyen adott i-nél egy $< v_1 \dots v_l >$ legyező és a hozzá tartozó cd-út. Mivel i-nél a c szín szabad, ezért a cd-út első, (i, v) éle d színű (ha az út nem 0 hosszú), és $v = v_k$ valamely k-ra, azaz v benne van a legyezőben (különben nem lenne maximális a legyező).

Az algoritmus a következő módon színezi az (i, v_1) élt:

- Keres egy $< v_1 \dots v_l >$ maximális legyezőt.
- Keres egy cd-utat. (A c, d színekre általában többféle választási lehetőség van. Egy mindig van, mert $D + 1$ színnel dolgozunk.)
- meghatároz egy w pontot, melyre
  - $w = v_l$ ha egy legyező-él sem d színű VAGY a v_{k-1} pont benne van a cd-útban.
  - $w = v_{k-1}$ ha a v_{k-1} pont nincs benne a cd-útban.
- A cd-utat invertálja.
- Ekkor $< v_1 \dots w >$ egy (nem feltétlenül maximális) legyező lesz (ezt az állítást később belátjuk), ezt megforgatja.
- Legyen az (i, w) él színe d.

Azt kell csak látni, hogy $< v_1 \dots w >$ tényleg egy legyező, és hogy az invertálás után is szabad marad w-nél a d szín (azaz az utolsó lépés sem rontja el a színezés helyességét).

- Ha egy legyező-él sem volt d színű, akkor a cd-út 0 hosszú, nincs invertálás, $w = v_l$, az eredeti legyezőnket választottuk. $w = v_l$-nél a d szín persze szabad marad.
- Ha a v_{k-1} pont nincs benne a cd-útban ($w = v_{k-1}$), akkor nála az invertálás után is szabad marad a d szín. Tehát a $< v_1 \dots v_{k-1} >$ egy legyező lesz, mert az invertálás után is igaz marad, hogy az (i, v_{j+1}) él színe szabad a v_j pontnál ($j = 1, \dots, k-2$).

- Ha a v_{k-1} pont benne van a *cd*-útban, akkor nála végződik a *cd*-út egy c színű éllel (mert v_{k-1}-nél eredetileg szabad volt a d szín). Az invertálás után nála szabad lett a c szín, és pont erre van szükség, mert az (i, v_k) él színe c lett. A v_l nincs benne a *cd*-útban (mert csak a másik végpontja lehetne, de $v_{k-1} \neq v_l$), ezért nála szabad maradt a d szín. Tehát a $< v_1 \dots v_l >$ tényleg egy legyező.

Ezt a színezési eljárást iteráljuk az egyes pontoknál, amíg minden él meg nem lesz színezve.

Megjegyzés: Amíg lehet, az egyes lépéseknél érdemes a c, d színeket úgy választani, hogy $c = d$ legyen, mert ekkor nincs szükség az időigényes *cd*-út invertálásra, és így gyorsabb lesz az algoritmus.

2.3.3. Élszínezés

Misra és Gries algoritmusa centralizált. Ezt az egyes pontoknál különkülön alkalmazva ($D = \Delta_2$-vel), egy elosztott algoritmust kaphatunk G_2 élszínezésének elkészítésére. Szükség lesz azonban néhány kiegészítésre, hogy az elosztott megvalósítás során felmerülő problémákat elkerüljük.

Minden i pont a saját, szomszédos éleit színezi. Egy él színezéséhez készít egy legyezőt. Ahhoz, hogy ez a legyező ne változzon meg menet közben, az szükséges, hogy az i szomszédainál levő éleken ne változzanak a színek. Tehát ha az i pont színez, akkor a tőle legfeljebb 2 távolságra levő pontok nem színezhetnek.
A *cd*-út invertálásához pedig az kell, hogy az út élei ne váltsanak menet közben színt.

A színezést két menetre bontjuk. Csak a második menetben fogunk használni *cd*-utakat.

Első menet

$\mathcal{C}$ = a színek halmaza
$\mathcal{C}_i$ = az i-nél levő élek színeinek halmaza (egy adott pillanatban)
$\mathcal{F}_i = \mathcal{C} \backslash \mathcal{C}_i$

Az i pont csak azután kezdi el színezni az éleit, miután az összes, tőle legfeljebb 2 távolságra levő, nagyobb ID-jű pont ezt befejezte. Annyi élet színez ki, amennyit tud, a következő módon. Begyűjti minden k szomszédjától a megfelelő $\mathcal{C}_k$ értékeket. Kiválasztja valamelyik (i, v_1) színezetlen élt.

Elkészít egy $< v_1 ... v_l >$ legyezőt, majd a c, d színeket úgy választja, hogy ha $\mathcal{F}_i \cap \mathcal{F}_{v_l} \neq \emptyset$, akkor $c \in \mathcal{F}_i \cap \mathcal{F}_{v_l},\ d = c$. Ekkor a cd-út 0 hosszú lesz, azaz nincs szükség a cd-út invertálására. A legyezőt megforgatja, és a színezetlen (i, v_1) élnek a d színt adja.

Ugyanígy tesz a többi színezetlen élnél is, amíg a kapott legyezőben tud úgy választani, hogy $c = d$ legyen. Ha már nem, akkor közli a szomszédjaival a megszínezett élek színét, majd a szomszédokat és másodszomszédokat tájékoztatja, hogy befejezte az első menetet.

Második menet

A cd-utak invertálása nem feltétlenül történhet egyszerre, mert ha két ilyen útnak van közös színe, akkor az utak metszhetik egymást. Ezért zárakat fogunk kérni a cd-utakon az élekre az egyes csúcsoknál, és vigyázunk, hogy elkerüljük a „deadlock" szituációkat, azaz az olyan helyzeteket, amikor a cd-utak egymásra várnak.

A zárakra bevezetünk *prioritásokat* (rendezést). Ha egy i pontból induló cd-út keresése közben kérünk zárat, akkor ennek a zárnak a prioritása az i pont ID-je lesz. Minél nagyobb ez az ID, annál nagyobb a kérés prioritása.

Kétféle zárat használunk: *elővételi és nem-elővételi zárat.*

Az i pont elindítja a cd-út keresést: tesz egy elővételi zárat a belőle induló d színű élre, majd továbbítja a zár-kérést az él másik (j) végpontjához. A cd-út további pontjai elővételi zárakat tesznek a hozzájuk csatlakozó c, d színű élekre, majd továbbítják az úton a zár-kérést. Az út végpontja az utolsó élre tesz egy nem-elővételi zárat, majd visszaküldi az úton az útkeresés sikerességét. A közbülső pontok cserélik az elővételi zárakat nem-elővételiekre. (Mire az üzenet visszaér i-hez, a cd-út összes élére nem-elővételi zárak vannak az út pontjainál.) Ekkor i üzen az út pontjainak, hogy cseréljék az élek színét és oldják fel a zárakat.

Ha egy pont kap egy kérést, hogy tegyen elővételi zárat egy élre egy cd_1-úton, de az élen már van egy másik, valamely cd_2-úthoz tartozó zár, akkor csak abban az esetben cseréli le a zárat, ha a cd_2-es út zárja elővételi és prioritása kisebb, mint a cd_1-esé. Ekkor cseréli a zárat, és a cd_2-úton mindkét irányban küld egy üzenetet az útkeresés sikertelenségéről. Ennek hatására a cd_2-úton a zárak feloldódnak, és az út kezdőpontja újrakezdi az út felépítését.

Amennyiben viszont nem ez a helyzet, akkor a cd_1-es kérésnek (és egyben a cd_1-út keresésnek) várnia kell, amíg a zárat feloldják.

Mielőtt egy pont színezne a második menetben, nem-elővételi zárakat tesz a belőle induló élekre magánál és a szomszédjainál. Ennél a lépésnél egy kis különbséget kell tenni a zár-kezelésben az előzőekhez képest.

Tekintsük azt az esetet, amikor egy i pontnak valamelyik élre (magánál vagy egy szomszédjánál) nem sikerül zárat tenni ebben a végső szakaszban, mert az élen már van egy L zár.

- Ha az L zár prioritása nagyobb, mint az i ID-je, akkor (függetlenül attól hogy az L zár elővételi-e vagy sem) törölni kell az összes, az i által ebben a záró szakaszban lefoglalt zárat, és az i-ből induló cd-úton lefoglalt zárakat is. Ezek a törlendő zárak mind nem-elővételiek. (Ez a zár-törlési művelet implementálható elosztottan, de itt eltekintünk ennek részletezésétől.) Ekkor az i pontnak természetesen újra kell indítania a cd-út keresést.

- Ha az L zár prioritása kisebb, mint az i ID-je: amennyiben az L elővételi, akkor törölhető, ha pedig nem-elővételi, akkor az i által indított zár-kérésnek várni kell, amíg az L zárat feloldják (ez a két eset tehát a fentebb leírt protokolltól nem tér el).

Ezzel a módszerrel elkerülhetők a deadlock szituációk, amikor az egyes zár-kérések körben várják egymást. (Meggondolható: a legnagyobb prioritású cd-út keresés és színezés előtti él-lefoglalás egy idő után mindenképpen megtörténik.)

Az élszínezést tehát elkészítettük. Most következik az él-ütemezés.

2.3.4. Az él-ütemezés elkészítése

Az egyszerűség kedvéért egy centralizált leírást adunk, mely könnyen implementálható elosztott módon a [13],[25] alapján.

$G_2(V, A_2)$ jelöli a hálózatot. Egy adott élszínezés esetén egy g színre $G_2^g(V^g, E^g)$ jelölje azt a gráfot, melyre V^g a g színű élek végpontjai, E^g pedig a V^g által feszített élek.

Minden g színnek két időegység fog megfelelni az él-ütemezésben. Ezekben csak a V^g-beli pontok fognak üzenetet küldeni vagy fogadni.

Tekintsük az első időegységet a kettőből. A V^g-beli pontoknak előjelet adunk, ahol a „+” előjel azt fogja jelenteni, hogy ebben az időegységben a pont csak küldhet, a „−” pedig azt, hogy csak fogadhat. Vagyis, ha egy „+” jelű pontból egy „−” jelű pontba mutat egy él az *eredeti* G hálózatban, akkor

közöttük ebben az időegységben történhet üzenetküldés. Ha G-ben olyan él (is) van, mely a két pont között éppen az ellenkező irányba mutat, akkor az az él a g-nek megfelelő második időegységhez fog tartozni.

Minden g színre, G_2^g-nek egy *érvényes előjelezése* olyan $V^g \to \{+,-\}$ hozzárendelés, melyre ha $x \in V^g$ jele $+/-$, akkor a x szomszédai közül csak annak lehet $-/+$ a jele, mely x-szel a g színű éllel van összekötve.

G_2^g-nek egy érvényes előjelezésénél tehát azok a pontok, melyek g színű éllel vannak összekötve, különböző előjelűek, a többi pontpár azonos előjelű.

Nyilvánvaló, hogy egy érvényes előjelezés „megfordítása" is érvényes, vagyis amelyben minden pontnak az előjelét az ellentettjére változtatjuk.

Megjegyzés: Ezen a ponton érdemes megfigyelni, hogy ha sikerül egy érvényes előjelezést készíteni az egyes G_2^g-khez, akkor az ezáltal kapott él-ütemezés eleget tesz az eredeti feladatunkban megadott korlátozásoknak. Egyrészt G_2 színezésének helyessége és az előjelezés miatt szomszédos G-beli éleken nem történhet üzenetküldés ugyanabban az időegységben; másrészt az előjelezést úgy készítettük, hogy az E_{tr}^1 korlátozásnak is eleget tegyen a kapott él-ütemezés (ez könnyen meggondolható).

Ha egyáltalán létezik érvényes előjelezés egy konkrét g színre, akkor G_2^g-t egy DFS bejárással meg lehet előjelezni. Ezt elosztottan úgy oldhatjuk meg, hogy indítunk külön-külön, minden pontból DFS-t, de ha egy pontot újra akarnánk előjelezni, akkor csak azt az előjelet tartjuk meg, amelyik a nagyobb ID-jű pont által indított DFS bejáráshoz tartozik. Ennek az lesz az eredménye, hogy végül csak a legnagyobb ID-jű pont DFS-e során készített előjelezés marad meg.

Könnyen látható, hogy ez az eljárás mindig ad egy érvényes előjelezést, ha G_2 egy fa. Ha kör is van G_2-ben, akkor benne a páros sok g színű élből álló körök előjelezhetők lesznek, a többi kör nem. (Mert minden g színű élen előjelváltás történik.)

Az is könnyen látható, hogy mivel DFS-t használtunk, ezért ha ezzel nem sikerült érvényes előjelezést csinálni, akkor az nem is létezik. Ugyanis ez a baj a DFS-fában csak egy „vissza-élen" való lépéskor következhet be.

Ha tehát a (legnagyobb ID-jű) $i \in V^g$ pontnak a DFS bejárás során nem sikerült G_2^g-t előjeleznie, akkor i a hozzá csatlakozó g színű élnek ad egy másik színt. (Ekkor lehet, hogy új színt kell bevezetnie a rendelkezésre álló $\Delta_2 + 1$ színen túl.) Ebben az esetben újraindul a kísérlet az előjelezés elkészítésére, az immár eggyel kevesebb élet tartalmazó G_2^g-ben.

2.3.5. Végeredmény fákra

Ha G_2 egy fa volt, akkor tehát legfeljebb $\Delta_2 + 1$ színt használtunk fel és mindegyik színhez pontosan 2 időegységet, összesen tehát legfeljebb $2(\Delta_2+1)$ időegységből kell állnia az él-ütemezésben meghatározandó idő-szakasznak.

G_2 konstrukciójából nyilvánvaló, hogy $\Delta_2 \leq \Delta \leq 2\Delta_2$, így tehát a szükséges idő-szakasz hossza legfeljebb $\Delta + 2$.

3. fejezet

Alvásidő-ütemezés

3.1. Bevezetés

Egy szenzorhálózatban levő szenzorok valamilyen lokális áramforrással működnek, ami lehet például egy elem vagy akkumulátor. A szenzorok általában nem túl nagyok, ezért az áramforrásnak is viszonylag kis méretűnek kell lennie. Az áramforrásról általában tudni lehet, illetve kiszámítható, hogy ha a szenzor folyamatosan használja, akkor az mennyi ideig tud áramot szolgáltatni. Ez a működési idő az elem vagy akkumulátor kis méretéből adódóan gyakran igen rövid lehet. Ha azt szeretnénk, hogy a hálózatunk hosszabb ideig működhessen úgy, hogy közben ne kelljen a szenzorokban elemet cserélni, vagy az akkumulátorokat újratölteni, akkor nem tehetjük meg, hogy minden szenzor folyamatosan bekapcsolt állapotban legyen. Szükség van arra, hogy a szenzoroknak megengedjük, hogy „aludjanak", sőt, az idő nagyrészében kikapcsolva legyenek (vagy készenléti, energiatakarékos állapotban), és csak néha-néha kapcsolják be magukat, és lépjenek működésbe. Ezzel a módszerrel jelentősen meghosszabbíthatjuk a hálózat zavartalan működésének időtartamát.

Felmerül azonban egy probléma. Ha a hálózatban szükséges, hogy a szenzorok üzeneteket küldjenek egymásnak, akkor ha ügyetlenül határozzuk meg, hogy az egyes szenzorok mely időegységekben legyenek ébren és mikor aludjanak, akkor a köztük menő üzenetek sokat késhetnek ahhoz képest, mintha minden szenzor folyamatosan ébren lenne. Ez pedig nyilván ellehetetleníti az egész hálózat működését. Szükség van tehát a szenzorok alvásidejének hatékony ütemezésére, ahol hatékonyság alatt azt értjük, hogy a szenzorok között menő üzeneteknek ne legyen nagy a késésük.

Feltételezések, megjegyzések

Az alvásidő-ütemezésnek olyan hálózatokban van igazi jelentősége, melyekben átlagosan kicsi a kommunikációs igény. Ugyanis ha a hálózatban viszonylag sok üzenetnek kell áthaladnia, vagyis a szenzorok az egyes időintervallumokban átlagosan sok üzenetet küldenek, akkor ez azt jelenti, hogy az alvásidő-ütemezésnek köszönhetően ugyan meghosszabbítjuk a szenzorok élettartamát, viszont az üzenetek túl sokat fognak késni ahhoz képest, amennyi energiát a szenzorok megtakarítanak. Akkor érdemes lehetővé tenni a szenzorok számára az alvást, ha enélkül az idő nagyrészében ugyan be lennének kapcsolva, de nem vennének részt az üzenetek továbbításában. – Mindenesetre, a későbbiekben ezzel a kérdéssel nem fogunk foglalkozni, azaz nem fogjuk vizsgálni, hogy a szenzorok átlagosan hány időegységenként küldenek üzenetet a szomszédaiknak. Célunk mindössze az lesz, hogy üzenetek késését minimalizáljuk.

Ebben a fejezetben már nem foglalkozunk azzal a kérdéssel, hogy milyen feltételeknek kell eleget tennünk az üzenetek küldésekor. Feltesszük, hogy bármely szenzor bármely szomszédjának tud üzenetet küldeni, a címzett pedig a küldeményt képes fogadni, amennyiben éppen ébren van.

Az üzenetek adott hosszúságúak. (Azonos hosszú üzeneteket készíthetünk például az üzenetek feldarabolásával.) Ha különböző hosszúságú üzeneteket is lehetne küldeni, akkor a leghosszabb lehetséges üzenet elküldéséhez szükséges időt kellene egységnek választani, ami jelentősen csökkentené az energiamegtakarítást.

A szenzorok alvásidő-ütemezésével azt fogjuk meghatározni, hogy mely időegység(ek)ben fogadhatnak üzenetet, míg a küldésre majd bármelyik időegységben sort keríthetnek. Ha egy szenzornak egy üzenetet el kell küldenie valamelyik szomszédjának, akkor ezt abban az időegységben fogja megtenni, amelyikben a kérdéses szomszéd azt fogadni tudja. (Ez az információ a küldő szenzor számára valamilyen formában rendelkezésre áll.)

Előfordulhat, hogy a hálózatban az alvásidő-ütemezés után két szenzor között nem a legrövidebb úton lesz a legkisebb az üzenet késése, ami azt eredményezi, hogy a kisebb késés érdekében több energiát kell befektetni egy üzenet kézbesítéséhez. Ezt azonban nem tekintjük problémának, hiszen az energiamegtakarítás nagyrésze nem az üzenetek minél rövidebb úton való kézbesítésének, hanem a szenzorok alvásának köszönhető.

3.2. A feladat meghatározása

A szenzorhálózatot egy $G = (V, E)$ irányítatlan gráffal modellezzük, ahol V a szenzorok halmaza, és két szenzor között akkor van él, ha tudnak egymásnak üzenetet küldeni. Ebben a modellben tehát feltételezzük, hogy ha egy v szenzornak a hatótávolságán belül van egy w szenzor, akkor ez a viszony kölcsönös.

Adott egy $k \in \mathbb{N}$ szám, hogy a szenzoroknak egy időintervallum időegységeinek legfeljebb $1/k$-adrészében szabad ébren lenniük. (Ezzel a paraméterrel szabályozható, hogy milyen gyakran legyenek aktívak a szenzorok.)

Adott egy $t \in \mathbb{N}$ szám, hogy az egy időintervallumban levő időegységek száma tk.

Alvásidő-ütemezést készítünk: minden ponthoz hozzárendelünk legalább egy, de legfeljebb t darab $\{0, \ldots, tk-1\}$-beli értéket. (Ez azt jelenti, hogy a pontnak megfelelő szenzor ezekben az időegységekben fogadhat üzenetet.) Jelölje ezt a hozzárendelést $f : v \in V \to \mathcal{F}_v \subseteq \{0, \ldots, tk-1\}$.

Egy $(v, w) \in E$ *élen küldött üzenet késése*, amennyiben v-ből w-be megy a küldemény:

$$d_f(v,w) = \begin{cases} tk & (\text{ha } \mathcal{F}_v = \mathcal{F}_w \text{ és egyeleműek}) \\ \min\{(f_w - f_v) \pmod{tk} : & \\ \quad f_v \in \mathcal{F}_v, f_w \in \mathcal{F}_w, f_v \neq f_w\} & (\text{különben}) \end{cases}$$

Jelölje $d_f(P) = \sum_{(v,w)\in P} d_f(v,w)$ a P *(irányított) úton a késést*, valamint $d_f(v,w)$ a $v-w$ *(irányított) utakon levő késések minimumát.* Ez utóbbi mennyiség szomszédos v és w pontokra megegyezik a fentebb definiált, (szintén $d_f(v,w)$-vel jelölt) élen küldött üzenet késésével.

1. változat

Mindegyik pont mindegyiknek ugyanakkora valószínűséggel akar üzenetet küldeni.

Legyen $D_f = \max\{d_f(v,w) : v, w \in V, v \neq w\}$ a *késési átmérő.*
Feladat: olyan alvásidő-ütemezést készíteni, azaz olyan f-et találni, amire a D_f késési átmérő minimális.

2. változat

Más-más az üzenetküldés valószínűsége az egyes pontpárok között. Ekkor a feladat a várható késési idő minimalizálása lesz.

Legyen $P(v,w)$ az üzenetküldés valószínűsége v-ből w-be,
$D_f^{avg} = \sum_{v,w\in V} P(v,w) d_f(v,w)$ az *átlagos késési átmérő.*

Feladat: olyan alvásidő-ütemezést készíteni, azaz olyan f-et találni, amire a D_f^{avg} átlagos késési átmérő minimális.

NP-teljesség

Annak eldöntése, hogy adott G, k, f-re és adott D számra igaz-e, hogy $D_f \leq D$, illetve $D_f^{avg} \leq D$, már a $t = 1$ esetben is NP-teljes. [17], [18]

A továbbiakban csak az 1. változat feladatával foglalkozunk, a [17] cikket kifejtve. Speciális gráfokra vizsgáljuk meg a problémát, és adunk optimális alvásidő-ütemezést vagy közelítő megoldást. A fejezet végén felvázolunk egy módszert arra az esetre, amikor tetszőleges gráffal van dolgunk.

3.3. A $t = 1$ eset fán és körön

A $t = 1$ esetben az időintervallum k időegységből áll, és az f leképezés egy $V \to \{0, \ldots, k-1\}$ függvény lesz; jelöljük minden $v \in V$-re az egyelemű $\mathcal{F}_v$ halmazt $f(v)$-vel.

Tehát egy $(v, w) \in E$ élen küldött üzenet késése, amennyiben v-ből w-be megy a küldemény:

$$d_f(v,w) = \begin{cases} k & (\text{ha } f(v) = f(w)) \\ (f(w) - f(v)) \pmod k & (\text{ha } f(v) \neq f(w)) \end{cases}$$

Ebből következik:

$$d_f(v,w) + d_f(w,v) = \begin{cases} 2k & (\text{ha } f(v) = f(w)) \\ k & (\text{ha } f(v) \neq f(w)) \end{cases}$$

3.3.1. Fák

11. Tétel. *Legyen $T = (V, E)$ egy fa, k az időegységek száma. Jelölje T átmérőjét h. Ekkor minden $f : V \to \{0, \ldots, k-1\}$ alvásidő-ütemezésre $D_f \geq \frac{hk}{2}$.*

Bizonyítás: Jelölje $a \in V$ és $b \in V$ azt a két pontot, melyek között h felvétetik. Tekintsük valamely p és q pontok között a fában az egyértelmű P utat, ennek hosszát jelölje s. (Egy *út hosszán* az éleinek a számát értjük.)

Legyen f egy tetszőleges alvásidő-ütemezés.

$$d_f(p,q) + d_f(q,p) = \sum_{(v,w)\in P} d_f(v,w) + \sum_{(v,w)\in P} d_f(w,v)$$
$$= \sum_{(v,w)\in P} (d_f(v,w) + d_f(w,v)) = sk.$$

Behelyettesítve az a, b pontpárt, azt kapjuk, hogy $d_f(a,b)+d_f(b,a) = hk$. Így $D_f \geq \max\{d_f(a,b), d_f(b,a)\} \geq \frac{hk}{2}$. $\square$

Optimális megoldás fán

Adjuk meg a fa egy 2-színezését. Az egyik színosztály pontjaihoz rendeljük a 0 időegységet, a másik színosztály pontjaihoz a $\lceil \frac{k}{2} \rceil$ időegységet. Így tetszőleges p, q pontokra, melyeknek s a távolsága,

$$D_f = \max_{p,q\in T}\{\max\{d_f(p,q), d_f(q,p)\}\} = \max_{p,q\in T} \left\lceil \frac{xk}{2} \right\rceil = \left\lceil \frac{hk}{2} \right\rceil,$$

tehát erre az f-re a tétel szerint D_f minimális.

3.3.2. Körök

12. Tétel. *Tekintsünk egy $n = mk$ pontból álló C kört, a pontokat jelöljük $v_0, \ldots, v_{mk-1}$-gyel. Ekkor minden f'-re $D_{f'} \geq m(k-1)$, és az $f(v_i) = i \pmod k$ $(i = 0, \ldots, mk-1)$ hozzárendelésre egyenlőség áll, vagyis ez az f egy optimális alvásidő-ütemezés.*

Bizonyítás: A $k = 2$ eset triviális, tehát feltehetjük, hogy $k \geq 3$. A tételben megadott f-re nyilván $D_f = m(k-1)$.

Indirekt tegyük fel, hogy létezik $f' : D_{f'} < D_f$. Tekintsünk a v_0 és a v_m közötti utat a körön. Mivel $D_{f'} < m(k-1)$, ezért a v_0 és v_m közötti legkisebb késésű út (mindkét irányban) ez az út lesz, ugyanis a másik útnak $m(k-1)$ éle van, mindegyiken legalább 1 a késés. Ez igaz a kör minden m hosszú útjára.

A $v_{km} := v_0$ jelölést használva, $i = 1, \ldots, k$-ra $d_{i1} := d_{f'}(v_{(i-1)m}, v_{im})$, $d_{i2} := d_{f'}(v_{im}, v_{(i-1)m})$, ezek tehát mindig a megfelelő két pontot összekötő m hosszú úton vétetnek fel.

Állítás: $d_{min} = \min\{d_{ij} : i = 1, \ldots, k,\ j = 1, 2\} < 2m$.

Bizonyítás: Tekintsük a v_0 és a $v_{\lceil \frac{k-1}{2} \rceil m}$ közötti két utat.

A $v_0 \to v_m \to v_{2m} \to \ldots \to v_{\lceil \frac{k-1}{2} \rceil m}$ úton legalább $\lceil \frac{k-1}{2} \rceil\, d_{min}$,

a $v_0 \to v_{mk-m} \to \ldots \to v_{\lceil \frac{k-1}{2} \rceil m}$ úton legalább $\left(k - \lceil \frac{k-1}{2} \rceil\right) d_{min} = \lceil \frac{k}{2} \rceil d_{min}$ a késés. Tehát ha $d_{min} \geq 2m$ volna, az ellentmondana a $D_{f'} < m(k-1)$ feltevésnek. $\square_{\text{Állítás}}$

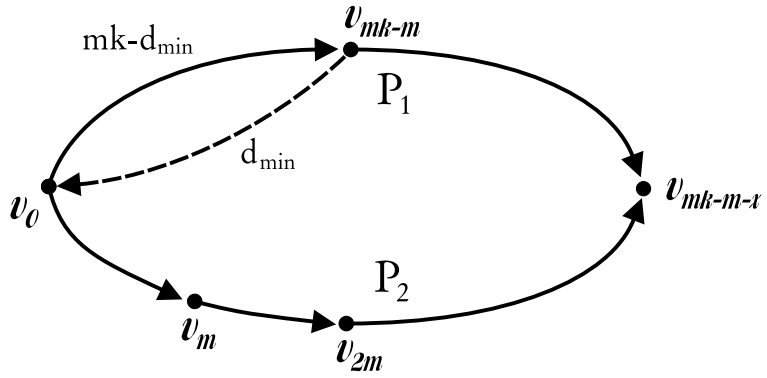
Mivel egy m hosszú úton legalább m a késés, ezért $m \leq d_{min} < 2m$. Legyen $d_{min} = m + x$, ahol $x \in \{0, \ldots, m-1\}$. Vétessen fel d_{min} például a v_{mk-m} és a v_0 között (azaz $d_{min} = d_{k1}$). Vizsgáljuk meg a $d_{f'}(v_0, v_{mk-m-x})$ mennyiséget (3.1. ábra).

Legyen $P_1 : v_0 \to v_{mk-m} \to v_{mk-m-x}$, erre a P_1 útra $d_{f'}(P_1) \geq mk - d_{min} + x = m(k-1) > D_{f'} \geq d_{f'}(v_0, v_{mk-m-x})$.

Legyen $P_2 : v_0 \to v_m \to v_{2m} \to \ldots \to v_{mk-m-x}$, erre a P_2 útra $d_{f'}(P_2) \geq \sum_{i=1}^{k-2} d_{i1} + (m-x) \geq (k-2)d_{min} + m - x = (k-2)(m+x) + m - x = m(k-1) + x(k-3) \geq m(k-1) > D_{f'} \geq d_{f'}(v_0, v_{mk-m-x})$ (ugyanis $k \geq 3$).

Azt kaptuk, hogy $\min\{d_{f'}(P_1), d_{f'}(P_2)\} > d_{f'}(v_0, v_{mk-m-x})$, ám itt definíció szerint egyenlőségnek kellene állnia. Ellentmondás.

Tehát az indirekt feltevés hamis, azaz minden f'-re $D_{f'} \geq D_f = m(k-1)$. A tételt beláttuk. $\square$



3.1. ábra. A két út v_0 és v_{mk-m-x} között

13. Tétel. *Az $n = mk + t$, $0 < t < k$ pontból álló körre az $n = (m+1)x + y$ jelöléssel tetszőleges f' alvásidő-ütemezésre*

$$D_{f'} \geq (m+1)k - \left\lfloor \frac{(m+1)k - y}{x} \right\rfloor .$$

A 12. Tétel-beli szekvenciális f alvásidő-ütemezésre itt $D_f = (m+1)(k-1)$ a késési átmérő (ami nem sokkal tér el az optimumra adott alsó korláttól).

A tétel bizonyítása a [17]-ben megtalálható. Hasonló az iménti bizonyításhoz, csak több számolást igényel.

3.4. A $t = 2$ eset fán és rácson

3.4.1. Fák

14. Tétel. *Jelöljük ki a $T = (V, E)$ fa egy r pontját gyökérnek. Jelöljük egy x pont távolságát a gyökértől $l(x)$-szel. Az x pont azokban a t időegységekben legyen ébren, melyekre $t - l(x)$ vagy $t + l(x)$ osztható $2k$-val ($t = 0, \ldots, 2k-1$). (Egy időintervallum tehát $2k$ hosszú lesz, a pontok legfeljebb 2 időegységben lesznek ébren.)*

Ennek az alvásidő-ütemezésnek a késési átmérője kisebb, mint $h+4k$, ahol h jelöli a fa átmérőjét.

Megjegyzés: Ennél sokkal jobbat nem várhatunk, még az ébrenlétek számának növelésével (vagyis a $t > 2$ esetben) sem, hiszen a késési átmérő mindenképpen legalább h.

Bizonyítás: Az üzenetek a következőképp továbbítódnak. Tegyük fel, hogy x akar üzenetet küldeni a tőle d távolságra levő y-nak, és ez a t időegységben jut eszébe. Legyen z az a pont a fában, mely az $x - y$ úton a legközelebb van r-hez. x vár addig a t_1 időegységig, melyre már $t_1 + l(x) \equiv 0 \pmod{2k}$. Ekkor elküldi az üzenetet az y-hoz vezető úton, r felé. Amíg az üzenet elér r-ig, addig minden élen 1 lesz a késés. Az $x - y$ úton t_2 a teljes késés. Amikor a küldemény megérkezett r-be, akkor r-nek várnia kell addig a t_3 időegységig, melyre már $t_3 - l(z) \equiv 0 \pmod{2k}$. Ekkor z továbbküldi az üzenetet y felé. A $z - y$ úton is minden élen 1 lesz a késés, és az üzenet végül a t_4 időegységben megérkezik. Tehát az üzenet teljes késése $t_4 - t = (t_1 - t) + (t_2 - t_1) + (t_3 - t_2) + (t_4 - t_3)$. Mivel $(t_4 - t_3) + (t_2 - t_1) = d$, és $t_1 - t \leq 2k - 1$, ill. $t_3 - t_2 \leq 2k - 1$ (ugyanis $t, t+1, \ldots, t+2k-1$ valamelyike osztható $2k$-val), ezért $t_4 - t \leq d + 4k \leq h + 4k$. □

3.4.2. Rácsok

15. Tétel. *Adott egy $G = (V, E)$ rács-hálózat, a rácspontokat jelöljük az (i, j) koordináta-párokkal ($x_1 \leq i \leq x_2$, $y_1 \leq j \leq y_2$, $x_1, x_2, y_1, y_2 \in \mathbb{Z}$). Legyen egy (i, j) koordinátájú pont ébren azokban a t időegységekben, melyekre $t + i, t - i, t + j, t - j$ valamelyike osztható $2k$-val ($t = 0, \ldots, 4k - 1$). (Egy időintervallum tehát $4k$ hosszú lesz, és egy pont legfeljebb 4 időegységben lehet ébren.)*

Ennek az alvásidő-ütemezésnek a késési átmérője kisebb, mint $h+8k$, ahol h jelöli a G átmérőjét ($h = (x_2 - x_1) + (y_2 - y_1)$).

Megjegyzés: Ismét az a helyzet, hogy ennél sokkal jobbat nem várhatunk, még a $t > 2$ esetben sem, hiszen itt is a késési átmérő legalább h kell, hogy legyen.

Bizonyítás: Tegyük fel, hogy valamely (i, j) üzenetet akar küldeni egy másik (p, q)-nak. Mutatunk egy utat köztük, hogy ezen az úton legfeljebb $h+8k$ lesz az üzenet késése. Nézzük például azt az esetet, amikor $i \leq p$, $j \leq q$ (a többi eset hasonlóan végiggondolható). Ekkor a két pont távolsága $d = p-i+q-j$. (i, j) vár addig a t_1 időegységig, melyre már $t_1 - i \equiv 0 \pmod{4k}$. Ekkor elküldi az üzenetet „vízszintesen", (p, j) felé. Amíg az üzenet elér (p, j)-ig, addig minden élen 1 lesz a késés. Ezen az úton $p - i$ a teljes késés. Amikor (p, i) megkapja az üzenetet, akkor vár addig a t_2 időegységig, melyre már $t_2 - j \equiv 0 \pmod{4k}$. Ekkor (p, j) továbbküldi az üzenetet „függőlegesen", (p, q)-ig, ezen az úton is minden élen 1 lesz a késés, összesen $q-j$. (i, j) és (p, j) is legfejlebb $4k-1$ időegységet vár, mielőtt továbbküldi az üzenetet, tehát az üzenet teljes késése kevesebb, mint $4k + (p - i) + 4k + (q - j) = d + 8k \leq h + 8k$. □

Megjegyzés: A bizonyításban megadott üzenetküldési eljárás természetesen könnyen implementálható is egy konkrét rács-hálózathoz.

3.5. Általános hálózatok

Fel fogjuk használni a következő tételt, mely a [2], [3] és [9] alapján megfogalmazható:

16. Tétel. *Legyen G egy tetszőleges n pontú hálózat, c elég nagy konstans. Jelölje $d_G(u, v)$ az u és v távolságát G-ben.*

Létezik $S \subseteq \{G$ feszítő fái$\}, |S| = c \log n$ úgy, hogy minden $u, v \in G$ pontpárra

$$\min\{d_T(u, v) : T \in S\} \leq d_G(u, v) \cdot c \log n,$$

ahol $d_T(u, v)$ jelöli az u és v távolságát T-ben.

17. Tétel. *Jelölje $l_T(x)$ az x pont távolságát valamely r gyökértől egy $T \in S$-ben. Az x pont legyen ébren azon t időegységekben, melyekre $t - l_T(x)$ vagy $t + l_T(x)$ osztható $2kc \log n$-nel valamely $T \in S$-re. Egy időintervallum tehát $2kc \log n$ hosszú lesz, és egy pont legfeljebb $2c \log n$ időegységben lehet ébren.*

Ennek az alvásidő-ütemezésnek a késési átmérője $O((h + k) \log n)$, ahol h jelöli a G átmérőjét.

Bizonyítás: Ismét megadunk egy üzenetküldési eljárást.

Ha például x küld üzenetet y-nak, azt úgy teszi, hogy megkeresi azt a T-t, melyre $d_T(x, y)$ minimális, majd ezen a fán továbbítja az üzenetet, a 3.4.1-ben tárgyalt módon. A 14. Tételből és a 16. Tételből adódik, hogy ennek az alvásidő-ütemezésnek a késési átmérője legfeljebb $d_T(x, y) + 4ck \log n = O((h + k) \log n)$. $\square$

4. fejezet

Lokalizáció

4.1. Bevezetés

Gyakran felmerül szenzorhálózatok kapcsán a helymeghatározás kérdése. Az volna a legegyszerűbb megoldás, ha minden egyes szenzort ellátnánk egy GPS készülékkel. Erre legtöbbször sajnos nincs lehetőség: egyrészt ez a megoldás igen költséges, másrészt a globális helymeghatározó rendszer csak szabadtérben telepített szenzorok esetén alkalmazható.

Többféleképp is lehet kezelni ezt a problémát, közülük most csak néhányról teszünk említést.

Az egyik ötlet az, hogy ugyan nem ismerjük egyetlen szenzor pozícióját sem, azt viszont könnyen meg tudjuk állapítani, hogy két szenzornak mekkora a távolsága – mármint azon szenzoroké, melyek belül esnek egymás látótávolságán. Ezen távolságadatok vizsgálatával kaphatunk bizonyos információkat a szenzorok pozíciójáról. A fejezet első részében bemutatunk egy ad-hoc algoritmust, mely ebben a helyzetben nyújt segítséget.

Úgy is lehet információt szerezni a szenzorok elhelyezkedéséről, hogy egy mozgó egység (legyen ez akár egy robot vagy egy dolgozó, aki GPS-szel a kezében adatokat gyűjt) bejárja azt a területet, ahol a szenzorok megtalálhatóak, közben távolságokat mér, koordinátákat számol. Ezzel a feladattal fogunk a fejezet második részében foglalkozni.

Végül, a harmadik részben arról a feladattípusról ejtünk szót, melyben néhány szenzor pozícióját ismertnek tekinthetjük (ezeket bázisoknak nevezzük), viszont nincs lehetőség a távolságmérésre: mindössze a szenzorok szomszédsági gráfja áll rendelkezésünkre, és ebből kell információt szereznünk az egységek pozíciójával kapcsolatban.

Megjegyzés: A lokalizáció feladata egy-, két- és háromdimenzióban is megfogalmazódik a különféle alkalmazásokban. Meg kell jegyeznünk azonban, hogy az egyes feladattípusoknál még egydimenzióban is legtöbbször NP-nehéz problémákkal kell szembenéznünk.

4.2. Bázis nélküli lokalizálás

4.2.1. A modell

Modellünk egy $G = (V, E)$ irányítatlan gráf, ahol V pontjai a szenzorok, és két pont között vezet él, ha a megfelelő szenzorok belül vannak egymás hatótávolságán. Pontosabban, azt feltételezzük, hogy ez a viszony eleve kölcsönös, mert minden szenzornak ugyanakkora (R) hatótávolsága van.

Ismertnek tekintjük minden szomszédos $n_i, n_j \in V$-re az n_i és n_j távolságát, amit $r_{i,j}$-vel jelölünk. (Ezek az adatok a szomszédos szenzorok méréseiből származnak. Megjegyezzük, hogy ezekben a számokban lehetnek mérési hibák, de ezzel most nem foglalkozunk.) Tehát n_i és n_j között van él G-ben, ha $r_{i,j} \leq R$.

Jelöljük $h_{i,j}$-vel az n_i és n_j között vezető legrövidebb út élszámát. (Az egyes $h_{i,j}$-k nem ismertek a modellben, azaz nem képezik részét a bemutatandó algoritmus inputjának. Azokat a $h_{i,j}$ értékeket, melyeket az algoritmus során használunk, egy Dijkstra szubrutinnal kiszámolhatjuk.)

4.2.2. A feladat

Keressük a pontoknak egy olyan beágyazását a síkba, mely konzisztens a mért $r_{i,j}$ értékekkel, vagyis ha két szenzor nincs messzebb egymástól R-nél, akkor a beágyazásban a távolságuk a megfelelő $r_{i,j}$ mennyiség.

Megjegyzések

- Ha létezik ilyen beágyazás a síkba, akkor ennek az elforgatásával, tükrözésével, illetve eltolásával szintén egy konzisztens beágyazást kapunk. (Ennek persze az az oka, hogy nincsenek bázis szenzorok.) Egyes alkalmazásokban azonban elegendő ez a fajta lokalizáció, mely a szenzorok pozícióját csak az egybevágósági transzformációktól eltekintve adja meg.

- Még az egybevágóságoktól eltekintve sem biztos, hogy egyértelműen létezik konzisztens beágyazás. Egy élhosszakkal rendelkező G *gráf*nak

egy (élhossztartó) $\mathcal{B}$ *beágyazása* a síkba *globálisan merev*, ha az egybevágósági transzformációktól eltekintve egyértelmű, azaz ha bármely más (szintén élhossztartó) beágyazásban a pontok közötti távolságok ugyanakkorák, mint $\mathcal{B}$-ben. (Ez a definíció megfogalmazható térbeli beágyazásokra is.) Egy beágyazás *generikus*, ha a benne a pontok általános helyzetűek. A G *gráf* 2D-ben/3D-ben (generikusan) *globálisan merev*, ha minden generikus beágyazása a síkba/térbe globálisan merev. Egy gráfról eldönteni, hogy globálisan merev-e, 2D-ben polinomiálisan megoldható feladat [11],[14], 3D-ben viszont nem ismert alkalmas teszt a globális merevségre.

- Feladatunk NP-nehéz, még abban a speciális esetben is, ha a G gráf globálisan merev [23],[27]. Tehát nem várhatunk optimális megoldást adó, P-beli algoritmust, azaz olyat, mely tökéletesen konzisztens lenne az előzetes távolságadatokkal – az ezektől vett eltérések négyzetösszegét szeretnénk minimalizálni.

4.2.3. Az AFL algoritmus

A [20] cikk alapján bemutatjuk az AFL algoritmust (a rövidítés az Anchor-Free Localization kifejezésből ered), mely a fenti feladatnak egy közelítő megoldását adja, vagyis a beágyazásbeli távolságok és az $r_{i,j}$ értékek nem fognak megegyezni, de nem sokkal térnek el egymástól. A cikkben csak szimulációs eredmények szerepelnek a közelítés mértékét illetően: összehasonlítva más lokalizáló algoritmusokkal, látható, hogy ez az eljárás messzemenően jobb eredményeket produkál. Ezt többek között azzal magyarázzák, hogy míg más eljárások gyakran növekvő módon építik fel a beágyazott gráfot, addig az AFL algoritmusban az egyes pontok koordinátáit egyszerre számoljuk ki, és iteratíve javítunk a kapott beágyazáson.

Az AFL algoritmus két fázisból épül fel. Az első fázisban kinevez egy pontot középpontnak, és ennek, valamint másik négy pontnak a segítségével felállít egy koordinátarendszert. Ebben a koordinátarendszerben megadja a többi szenzor polárkoordinátáinak kezdőértékét. A második fázisban lokális javításokat alkalmaz mindaddig, amíg a beágyazás konzisztenciája lényegesen javul.

Első fázis

1. Tetszőleges n_0 pontot választunk, például a legkisebb ID-jűt. (A további pontok választásánál is, ha több lehetséges pont adódik, szintén választhatjuk a legkisebb ID-jűt.)

2. n_1 legyen az a pont, melyen a $\max h_{0,1}$ felvétetik, azaz n_1 legyen minél távolabb n_0-tól. (Heurisztika: n_1 a gráf „szélén" van.)

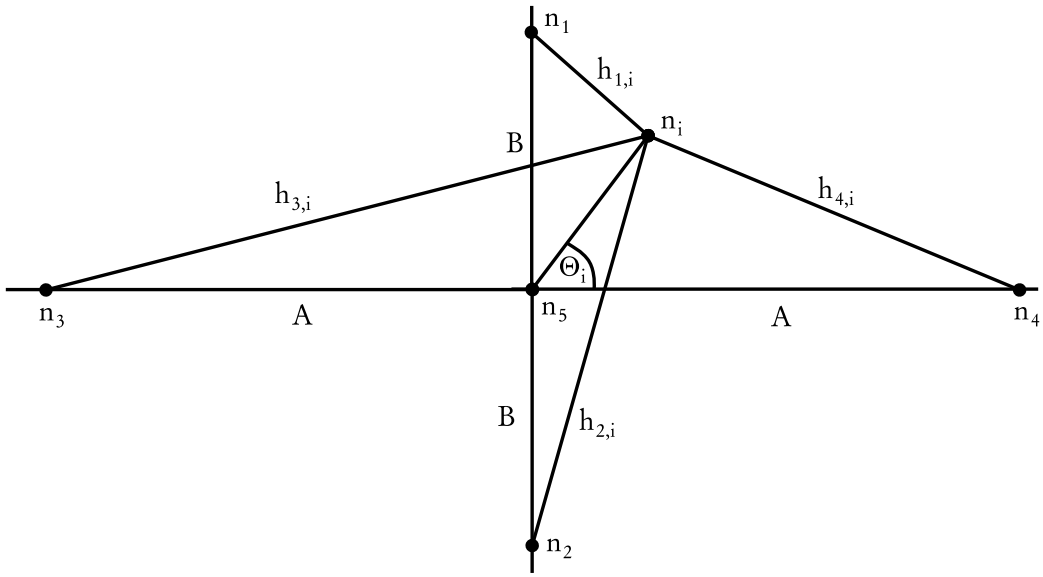
3. n_2 legyen az a pont, melyen a $\max h_{1,2}$ felvétetik, azaz n_2 legyen minél távolabb n_1-től, (n_2 a gráf „másik végében" van.)

4. n_3 legyen az a pont, ahol $\min |h_{1,3} - h_{2,3}|$ felvétetik, és ha több ilyen pont van, akkor vegyük azt, melyre $h_{1,3} + h_{2,3}$ maximális.

5. n_4 legyen az a pont, ahol $\min |h_{1,4} - h_{2,4}|$ felvétetik, ezen belül amire $h_{3,4}$ maximális. (n_3 és n_4 is a gráf két, átellenes végén van, merőlegesen az n_1 és n_2 által meghatározott „egyenesre")

6. n_5 legyen az a pont, ahol $\min |h_{1,5} - h_{2,5}|$ felvétetik, ezen belül amire $|h_{3,5} - h_{4,5}|$ minimális. (n_5 körülbelül a gráf „közepén" van.)

7. Meghatározzuk $n_1, \ldots, n_5$ segítségével a többi $n_i \in V$ pont kezdeti polárkoordinátáit: $\rho_i = h_{5,i}R$, $\Theta_i = \arctan \frac{h_{1,i}-h_{2,i}}{h_{3,i}-h_{4,i}}$.



4.1. ábra. Heurisztika az n_i pont Θ_i elforgatási szögének magyarázatához

Heurisztika a kezdeti polárkoordinátákhoz

A sugarat (az n_5 középponttól vett távolságot) a legrövidebb út hosszának R-szeresére választva egy olyan kezdeti állapotot kapunk, mintha a gráfunk egy *rugós modell* lenne: a pontok közötti élek a rugók, és ezt a szerkezetet a közepét (n_5-öt) rögzítve és attól minden irányban távolodva „szétfeszítenénk".

Az elforgatási szög választása a következőképp magyarázható (4.1. ábra). Ha az n_3 és n_4 pontok által meghatározott egyenest képzeljük egy koordinátarendszer x tengelyének, az n_1 és n_2 egyenesét az y tengelynek, n_5-öt az origónak; feltételezve továbbá, hogy $h_{3,5} = h_{4,5} = A$, illetve $h_{1,5} = h_{2,5} = B$, akkor kiszámolható, hogy a kapott koordinátarendszerben az n_i pont n_5 körüli elforgatási szögének tangense $\frac{(h_{1,i}-h_{2,i})(h_{1,i}+h_{2,i})A}{(h_{3,i}-h_{4,i})(h_{3,i}+h_{4,i})B}$, ahol a $\frac{(h_{1,i}+h_{2,i})A}{(h_{3,i}+h_{4,i})B}$ mennyiséget 1-nek feltételezve a Θ_i képletében szereplő tangens értéket kapjuk.

Második fázis

Maradva a rugós modell heurisztikájánál: amint a szétfeszített szerkezetet (a középpontot továbbra is rögzítve) elengedjük, az magától összeugrik, mégpedig úgy, hogy a pontok egyszerre elkezdenek a rájuk ható erők eredőjének irányába mozdulni. Ezt fogjuk ebben a fázisban megvalósítani.

A pontok mozgását apró részekre bontjuk, mégpedig úgy, hogy minden pillanatban kiszámoljuk a pontokra ható erők eredőit, és minden pontot egyszerre elmozdítunk a rá ható eredő erő irányába.

1. Minden n_i pontnak van egy aktuális $\hat{p}_i$ pozíciója. (Ez az első pillanatban az első fázisban kiszámolt érték.) A pontok elküldik a pozíciójukat a szomszédaiknak. Így minden pont kiszámolhatja a szomszédaitól való távolságát az aktuális beágyazásban. Az így kapott távolsága egy n_i-nek valamely vele szomszédos n_j-től $\hat{d}_{i,j}$. Természetesen az n_i pont tudja az n_j-től való kezdetben mért $r_{i,j}$ távolságát is.

2. Legyen $\hat{v}_{i,j}$ a $\hat{p}_i$-ból a $\hat{p}_j$ irányába mutató egységvektor. Ezzel a jelöléssel, $F_{i,j} = \hat{v}_{i,j}(\hat{d}_{i,j} - r_{i,j})$ az *n_i-re ható, $\hat{v}_{i,j}$ irányú erő.*

   Ha kiszámoljuk egy n_i pontnál a rá ható erőket, melyeket a szomszédjai fejtenek ki, és ezeket összeadjuk, megkapjuk az *n_i-re ható eredő erőt*:

$$F_i = \sum_{(i,j)\in E} F_{i,j}$$

3. Ha az eredő erőt minden pontra kiszámoltuk, akkor az összes n_i pontot egyszerre elmozdítjuk a rá ható F_i irányába. Az elmozdítás nagysága legyen $\frac{|F_i|}{2m_i}$, ahol m_i jelöli az n_i pont szomszédainak számát. Vagyis

az elmozdítás mértéke annál kisebb, minél több szomszédja van egy pontnak.

Ezután újra kiszámoljuk az aktuális távolságokat, az eredő erőket, elvégezzük az elmozdításokat. Ezt ismételjük mindaddig, amíg a beágyazás konzisztenciája lényegesen javul, vagyis amíg a szerkezet *energiáját* jelentő $\sum_{(i,j)\in E}(\hat{d}_{i,j} - r_{i,j})^2$ mennyiség legalább 0.1%-kal csökken.

Megjegyzés: Az algoritmus leírását többé-kevésbé elosztott módon adtuk meg. A szükséges kiegészítésekkel, valóban implementálható elosztottan.

4.3. Lokalizálás mozgó egységgel

4.3.1. A feladat

Adott egy $G = (V, E)$ szenzorhálózat a háromdimenziós térben, valamint az éleken az $r_{i,j}$ értékek. Határozzunk meg egy mozgó egység segítségével újabb távolság-értékeket a szenzorok között (bővítsük az E halmazt E'-vé) úgy, hogy a kapott $G = (V, E')$ gráf globálisan merev legyen (3D-ben).

Megjegyzések

- A megoldáshoz használható mozgó egység azon pontoktól való távolságát tudja megmérni egy pillanatban, melyek az aktuális helyzetéből láthatóak. (Lehetnek bizonyos akadályok, melyek gátolják a láthatóságot.)
- Amennyiben rendelkezésünkre áll a feladatnak egy megoldása, akkor a helymeghatározás feladata (ld. 4.2.2) megoldható bármely bázis nélkül lokalizáló algoritmussal, akár az előző részben bemutatott AFL algoritmusnak egy háromdimenziós változatával is.

4.3.2. Segédállítások

Jelöljük m_j-kel azokat a pontokat, ahonnan a mozgó egység egy adott pillanatban távolságokat mér. Könnyen belátható az alábbi négy állítás.

1. Lemma. *Egy gráf globálisan merev, ha úgy építettük fel, hogy négy, nem egy síkban elhelyezkedő pont klikkjéből indulva úgy adtunk hozzá újabb pontokat, hogy azokat mindig a már meglevő gráf legalább négy pontjával összekötöttük.*

2. Lemma. *Két pont (n_0, n_1) távolságát meghatározza az m_0, m_1, m_2 pontoktól való távolsága, ha az öt pont egy síkban van és m_0, m_1, m_2 kollineáris.*

3. Lemma. *Három, nem kollineáris pont (n_0, n_1, n_2) páronkénti távolságát meghatározza az $m_0, \ldots, m_5$ egy síkban elhelyezkedő pontoktól való távolsága, ha az m_i-k közül semelyik három nem kollineáris, és az n_i pontok mind az m_j pontok síkja felett vannak.*

4. Lemma. *Négy pont (n_0, n_1, n_2, n_3) páronkénti távolságát meghatározza az $m_1, \ldots, m_7$ pontoktól való távolsága, ha ezen 11 pont közül semelyik három nem kollineáris.*

Megjegyzés: A 2-4. Lemma közül a bemutatandó algoritmusban majd mindig azt fogjuk használni, amelyiknek a feltételeit valamilyen speciális plusz információnak köszönhetően biztosítani tudjuk. Gyakran ugyanis úgy képzeljük a konkrét valós helyzetet, hogy egy szobában vannak a szenzorok, és a mozgó egység egy ember, aki a szobában járkálva méreget. Így a 3. Lemma feltételeinek teljesüléséhez csak annyit kell tudnunk, hogy a szenzorok mind a plafonon vannak (nem egy egyenesen), a méréseket pedig az emberünk mindig ugyanabból a magasságból végzi. A 4. Lemmához pedig csak annyi kell, hogy a szenzorok különböző magasságban legyenek felszerelve, emberünk pedig járkálás közben mérjen, hogy a járásból adódó „imbolygás" következményeképp a mérések különböző magasságból történjenek. Ezek nem tényszerű biztosítékok, de a gyakorlatban működőképes feltevések.

4.3.3. Az algoritmus

A [21] cikkben szereplő algoritmust mutatjuk be a feladat megoldására. Az algoritmus két fázisból épül fel.

Első fázis

1. Keressünk négy pontot, melyeket a mozgó egység egy adott helyzetéből lát.

2. Mérjük meg a mozgó egység távolságát a négy ponttól, a mozgó egység hét különböző helyzetéből. Ezután a 4. Lemma segítségével határozzuk meg a négy pont egymás közötti távolságait.

3. Lokalizáljuk a négy pontot az 1. Lemma segítségével.

Második fázis

1. Keressünk egy pontot, melyet már lokalizáltunk, de még nem vizsgáltunk a második fázisban.

2. Keressünk ennek a pontnak a látótávolságán belül olyan helyeket, ahonnan a mozgó egység lát egy olyan pontot is, melyet még nem lokalizáltunk, illetve még további 0, 1 vagy 2 már lokalizált pontot.

3. Minden ilyen helyen számítsuk ki a mozgó egységgel a 2-4. Lemmák segítségével a távolságokat a vizsgált 2, 3 vagy 4 pont között. Lokalizáljuk a még nem lokalizált pontot az 1. Lemma segítségével, ha már van legalább négy távolságadatunk közte és a már lokalizált pontok között.

Megjegyzés: Az algoritmus olyan helyzetekben használható igazán, amikor vannak bizonyos előismereteink a szenzorok elhelyezkedéséről, illetve a mozgó egység mozgásának tulajdonságairól. Nincs elméleti garancia arra, hogy egy adott szenzorhálózatra az algoritmus működik (azaz mindig találunk a lemmák követelményeinek eleget tevő pontokat), de a gyakorlatban a korábbi megfontolások alapján az eljárás kivitelezhető.

4.4. Távolságmérés nélküli lokalizáció 1D-ben

A továbbiakban a helyzetmeghatározásnak azt az esetét vizsgáljuk, amikor pusztán a szenzorok szomszédsági gráfját ismerjük, és nincs lehetőség a szomszédos szenzorok közötti távolságok mérésére. Továbbá, vannak bázisok (bázis-szenzorok), melyeknek ismerjük a pozícióját. Végül, feltesszük, hogy azonos a szenzorok hatótávolsága.

Feladat: Adott egy szenzorhálózat szomszédsági gráfja. Határozzuk meg a nem-bázisok pozícióját.

Felmerülő kérdések: Tudunk-e találni elfogadható időben olyan megoldást, mely kielégíti azokat a feltételeket, melyeket a szomszédsági gráf előír? Meg tudunk-e adni erre egy gyors elosztott algoritmust? Tudunk-e adni olyan megoldást, mely (valamilyen értelemben) a lehető legjobban konzisztens a szomszédsági gráffal?

4.4.1. A modell

A szenzorhálózatot egy $G = (V, E)$ irányítatlan gráf írja le, ahol a V-beli szenzorok $\mathbb{R}^d$-ben helyezkednek el, és $(s, s') \in E$, ha s és s' tud küldeni üzenetet egymásnak (hatótávolságon belül vannak). G-ről feltesszük, hogy összefüggő.

$(s(1), \ldots, s(d))$ jelöli az $s \in V$ szenzor valódi pozícióját $\mathbb{R}^d$-ben.

Az $N(s)$ halmaz tartalmazza s-et és a szomszédjait, $h(s, s')$ jelöli s és s' távolságát, vagyis az élek számát a köztük levő, G-beli legrövidebb úton.

s és s' ekvivalens, ha $N(s) = N(s')$. Feltesszük, hogy nincsenek ekvivalens szenzorok. (Ha vannak, akkor soroljuk őket osztályokba, vegyünk egy-egy reprezentáns elemet, és azokkal dolgozzunk tovább.)

Megjegyzés: Modellünkben nincs megszorítás arra, hogy mely szenzorok között lehet egyszerre üzeneteket küldeni a hálózatban.

4.4.2. Az egydimenziós eset

Az általánosság megszorítása nélkül feltesszük, hogy a szenzorok $[0, 1]$-ben vannak, valamint egy-egy bázis-szenzor van 0-ban és 1-ben. Jelölje az s szenzor $[0, 1]$-beli pozícióját $p(s)$.

Feladatunk még egydimenzióban is nehéz, mert kevés információ áll rendelkezésünkre. Ezt mutatja az alábbi tétel.

18. Tétel. *Bármely $\mathcal{A}$ algoritmusra, mely a fent definiált feladatot egydimenzióban megoldja, létezik egy olyan $G = (V, E)$ inputgráf, melyben valamely $s \in V$ pontra $|p(s) - \mathcal{A}(s)| \geq \frac{1}{6}$, ahol $\mathcal{A}(s)$ az $\mathcal{A}$ becslése $p(s)$-re.*

Bizonyítás: Legyen $0 < r < 1$ a hatótávolság. $n = \frac{1}{r + \frac{r+\varepsilon}{2}}$ pontot teszünk le $[0, 1]$-be ($\varepsilon > 0$). Megadunk két gráfot (G' és G''), és egy s szenzor $[0, 1]$-beli pozícióját $p_{G'}(s)$-sel, illetve $p_{G''}(s)$-sel jelöljük.

$$p_{G'}(s) = \begin{cases} i \cdot r & i \leq \frac{n}{2} \\ \frac{nr}{2} + k(r + \varepsilon) + r & i = \frac{n}{2} + (2k + 1) \\ \frac{nr}{2} + k(r + \varepsilon) & i = \frac{n}{2} + (2k) \end{cases}$$

$$p_{G''}(s) = \begin{cases} k(r + \varepsilon) + r & i = 2k + 1 \leq \frac{n}{2} \\ k(r + \varepsilon) & i = 2k \leq \frac{n}{2} \\ \frac{n}{4}(r + \varepsilon) + r(i - \frac{n}{2}) & i \geq \frac{n}{2} \end{cases}$$

Ha ε elég kicsi, akkor G'-nek és G''-nek ugyanaz a szomszédsági gráfja, tehát $\mathcal{A}$-t futtatva ugyanazt az eredményt kapjuk. Látható, hogy $\varepsilon \to 0$

esetén a fenti $i = \frac{n}{2}$-hez tartozó s szenzornak G'-ben $\frac{2}{3}$, G''-ben $\frac{1}{3}$ lesz a pozíciója. Tehát a $G = G'$ vagy $G = G''$ választások egyikénél az iménti s-re $|p_G(s) - \mathcal{A}(s)| \geq \frac{1}{6}$. $\square$

A ranking feladat: Jelölje $rank_i(s) = |\{s' \in G : s'(i) \leq s(i)\}|$ az s szenzor i-edik rangját. A szenzorhálózat szomszédsági gráfjának ismeretében rendezzük a szenzorokat egy adott i-re $rank_i$ szerint.

Egydimenzióban ez a feladat a ($[0,1]$-ben levő) szenzorok rendezése, a szomszédsági gráf alapján, mindössze annyi előzetes információval, hogy van egy-egy szenzor a 0-ban és az 1-ben (ezeket B_0-lal és B_1-gyel jelöljük).

Most megadunk egy gyors elosztott algoritmust a [15] alapján, mely egydimenzióban megoldja a ranking feladatot.

Algoritmus az s nem-bázis pozíciójának becsléséhez:

1. Inicializálás: $\mathcal{A}_0(s) := 0$ ($\mathcal{A}_t(s)$ lesz az s-nek az algoritmus által becsült pozíciója a t-edik iteráció után), $t := 0$
2. $\mathcal{A}_{t+1}(s) := \frac{\sum_{j:s_j \in N(s)} \mathcal{A}_t(s_j)}{|N(s)|}$
3. s elküldi a szomszédainak $\mathcal{A}_{t+1}(s)$-et
4. Ha s megkapta $stop_k$-t ($k \in \{0,1\}$), akkor továbbküldi a szomszédainak (az algoritmus során csak egyszer)
5. Ha s már megkapta $stop_0$-t és $stop_1$-et is, akkor s algoritmusa leáll
6. $t := t + 1$, ugrás a 2. pontra

A B_i ($i = 0, 1$) bázis kezdetben elküldi a szomszédainak, hogy ő az i pozícióban van ($\mathcal{A}_0(B_i) = i$). Ha a kapott üzenetek alapján valamely t-re minden $s_j \in N(B_i)$-re $\mathcal{A}_t(s_j) > 0$, akkor B_i elküldi a szomszédainak $stop_i$-t.

19. Tétel. *Legyen s_1 a B_1-től legtávolabbi szenzor. $t = h(s_1, B_1)$ iteráció után a nem-bázisokra az $\mathcal{A}_t(s)$ ($s \in V$) értékek szerint a valóságnak megfelelő rendezést kapunk, azaz ha $p(s_i) < p(s_j)$, akkor $0 < \mathcal{A}_t(s_i) < \mathcal{A}_t(s_j) < 1$ minden $s_i, s_j \in V \backslash \{B_0, B_1\}$-re.*

5. Lemma. $0 \leq a_1 \leq a_2 \leq, \ldots, \leq a_i, \leq, \ldots, \leq a_{i+j}, \leq, \ldots, \leq a_{i+j+k} \Rightarrow \frac{\sum_{n=1}^{i+j} a_n}{i+j} \leq \frac{\sum_{m=i}^{i+j+k} a_m}{j+k}$ *és egyenlőség pontosan akkor van, ha* $a_1 = a_{i+j+k}$.

Jelölés: $\Gamma_i = \{s : h(s, B_1) \leq i\}$. Itt használjuk ki G összefüggőségét, mégpedig úgy, hogy minden $s \in V \backslash \{B_1\}$-hez létezik i, hogy $s \in \Gamma_i$.

6. Lemma. $\mathcal{A}_t(s) > 0 \Longleftrightarrow s \in \Gamma_t$.

Bizonyítás: t szerinti indukció.

Az 1. Lemmából következik, hogy az első iterációban csak B_1 szomszédjai becslik saját pozíciójukat 0-tól különböző helyre.

Tegyük fel, hogy valamely t-re igaz az állítás. Ekkor az 1. Lemmából adódóan minden $s \in \Gamma_t$-re $\mathcal{A}_{t+1}(s) > 0$. A Γ_{t+1}-en kívüli szenzorokra az indukciós feltevés és az 1. Lemma miatt $\mathcal{A}_{t+1}(s) = 0$. Az $s \in \Gamma_{t+1} \backslash \Gamma_t$-kre pedig $N(s) \cap \Gamma_t \neq \emptyset$ és ismét az 1. Lemma miatt $\mathcal{A}_{t+1}(s) > 0$. □

Következmény: B_0 a $stop_0$ jelet a $h(B_0, B_1) + 1$ időegységben küldi.

A 19. Tétel bizonyítása: A B_1-től való távolság (jel.: i) szerinti indukcióval megmutatjuk $\Gamma(i)$-re, hogy az i-edik iteráció után rendezett.

Figyeljük meg, hogy ha $s_j, s_k \in \Gamma_1$, $p(s_j) < p(s_k)$, akkor $N(s_k) \subsetneqq N(s_j)$, ezért az 1. Lemma alapján Γ_1 az első iteráció után rendezett.

Tegyük fel, hogy valamely $i < h(s_1, B_1)$-re az i-edik iteráció után Γ_i rendezett. Ekkor Γ_i az $i+1$-edik iteráció után is rendezett lesz, az indukciós feltevés és az 1. Lemma miatt. A 2. Lemma alapján minden $s_j \in \Gamma_{i+1}$-re $\mathcal{A}_i(s) = 0$, és minden $s_k \in \Gamma_i$-re $\mathcal{A}_i(s_k) > 0$, tehát az 1. Lemma miatt az $(i+1)$-edik iteráció után az egész Γ_{i+1} rendezett.

Mivel $i = h(s_1, B_1)$ esetén $\Gamma_i = V \backslash \{B_0\}$, ezért $h(s_1, B_1)$ iteráció után az egész gráf rendezett.

Az 1. Lemma alapján könnyen látható, hogy a nem-bázisok becsült pozíciói 0 és 1 között vannak. □

20. Tétel. *$t = 2h(B_0, B_1) - 1$ iteráció után az összes $s \in V$ algoritmusa leáll.*

Bizonyítás: B_1 az első iterációban az összes szomszédjától 0-tól különböző pozícióértéket kap, tehát a második iterációban elküldi a $stop_1$ jelet. A Következményből adódóan a $stop_0$ jel ennél később indul el, tehát a nem-bázisok a $stop_0$ jelet később (vagy legalábbis nem előbb) kapják meg, mint a $stop_1$-et, tehát a $stop_0$ jel érkezésekor fognak leállni. A $stop_0$ jelet legkésőbb a B_0-tól legtávolabbi szenzor fogja megkapni, ez a távolság pedig $h(s_1, B_1)$-gyel egyenlő. Emiatt és a Következmény miatt tehát minden $s \in V$ algoritmusa leáll $h(B_0, B_1) + h(s_1, B_1) = 2h(B_0, B_1) - 1$ iteráció után. □

Következmény: Egydimenzióban a ranking feladat (a fenti elosztott algoritmussal) az input méretében lineáris időben megoldható.

Bizonyítás: A 20. Tétel alapján a $G = (V, E)$ inputra az algoritmus futásideje $2h(B_0, B_1) - 1 \leq 2 \cdot |E|$. $\square$

Megjegyzés: Azt láttuk be, hogy az elosztott algoritmus az $\mathcal{A}(s)$ $(s \in V)$ értékek alapján $2h(B_0, B_1) - 1$ iteráció után a ranking feladatot megoldotta. Ez a rendezés egy globális információ, leálláskor az egyes szenzorok nem tudják, hogy ők hányadikok a sorban. Ahhoz, hogy ezt megtudják, az algoritmust ki kell egészíteni, de ez megoldható, és a futásidő lineáris marad.

Irodalomjegyzék

[1] Alekseev, V.B., and V.S. Gončakov. The thickness of an arbitrary complete graph. *Math. Sbornik* 30,2 (1976), 187-202.

[2] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. *37th IEEE Symposium on Foundations of Computer Science*, 184-193, 1996.

[3] Y. Bartal. On approximating arbitrary metrics by tree metrics. *30th ACM Symposium on Theory of Computing*, 1998.

[4] L.W. Beineke, F. Harary, and J.W. Moon. On the thickness of the complete bipartite graph. *Proc. Camb. Phil. Soc.* 60 (1964), 1-5.

[5] R. Connelly. On generic global rigidity. *Applied Geometry and Discrete Mathematics*, pp. 147-155, 1991.

[6] Alice M. Dean, Joan P. Hutchinson, and Edward R. Scheinerman. On the thickness and arboricity of a graph. *J. Combin. Theory Ser. B*, 52(1):147-151, 1991.

[7] M.B. Dillencourt, D. Eppstein, and D.S. Hirschberg. Geometric thickness of complete graphs. *J. Graph Algorithms and Applications* 4(3):5-17, 2000.

[8] Vida Dujmovič, David R. Wood. Graph Treewidth and Geometric Thickness Parameters. 2006.

[9] J. Fakcheroenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *Proceedings of ACM Symposium on Theory of Computing*, 2003.

[10] S. Gandham, M. Dawande, R. Prakash. Link scheduling in sensor networks: distributed edge coloring revisited. *IEEE Infocom*, March 2005.

[11] B. Hendrickson. Conditions for unique graph realizations. *SIAM Journal on Computing*, vol. 21, no. 1, pp. 65-84, 1992.

[12] http://home.mit.bme.hu/ nemeth/onlab/self-localization.html

[13] Isreal Cidon. Yet Another Distributed Depth-First-Search Algorithm. *Information Processing Letters*, 26(6):301-305, 1998.

[14] B. Jackson and T. Jordán. Connected rigidity matroids and unique realizations of graphs. *J. Combin. Theory Ser. B* 94(1):1-29, 2005.

[15] Z. Lotker, M. M. de Albeniz, and S. Perennes. Range-Free Ranking in Sensor Networks and Its Applications to Localization. *Proceedings of the Ad-Hoc, Mobile, and Wireless Networks: Third International Conference.* July 2004.

[16] Michael Jünger, Petra Mutzel, Thomas Odenthal, and Mark Scharbrodt. The thickness of a minor-excluded class of graphs. *Discrete Math.*, 182(1-3):169-176, 1998.

[17] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel. Delay Efficient Sleep Scheduling in Wireless Sensor Networks. *IEEE Infocom*, March 2005.

[18] Matthew J. Miller. An Errata for Delay Efficient Sleep Scheduling in Wireless Sensor Networks. *Technical Report*, September 2005.

[19] J. Misra and D. Gries. A Constructive Proof of Vizing's Theorem. *Inf. Proc. Lett.*, 41:131-133, 1992.

[20] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller. Anchor-Free Distributed Localization in Sensor Networks. *MIT Laboratory for Computer Science, Tech Report 892*, 2003.

[21] N. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller. Mobile-assisted localization in wireless sensor networks. *Proceedings of IEEE INFOCOM*, 2005.

[22] R. Ramanathan. A Unified Framework and Algorithm for Channel Assignment in Wireless Networks. *Wireless Networks*, 5:81-94, 1999.

[23] J. B. Saxe. Embeddability of weighted graphs in k-space is strongly NP-hard. *Proceedings of the 17th Allerton Conference on Communications, Control, and Computing*, 480-489, 1979.

[24] http://sensor.capsule.hu

[25] M. B. Sharma, N. K. Mandyam and S. S. Iyangar. An Optimal Distributed Depth-First-Search Algorithm. *Proceedings of the seventeenth annual ACM conference on Computer science : Computing trends in the 1990's*, 1989.

[26] V. G. Vizing. On an estimate of the chromatic class of a p-graph. *Diskret. Analiz.*, 3:25-30, 1964.

[27] Y. Yemini. Some theoretical aspects of position-location problems. *Proceedings of the 20th Annual IEEE Symposium on Foundations of Computer Science*, 1-8, 1979.