NYILATKOZAT

Név: Borsik Nóra Anna

ELTE Természettudományi Kar, szak: Matematika BSc

NEPTUN azonosító: N72R0F

Szakdolgozat címe:
Konszenzusos rangsorok

A **szakdolgozat** szerzőjeként fegyelmi felelősségem tudatában kijelentem, hogy a dolgozatom önálló szellemi alkotásom, abban a hivatkozások és idézések standard szabályait következetesen alkalmaztam, mások által írt részeket a megfelelő idézés nélkül nem használtam fel.

Budapest, 2022. május 30.

Borsik Nóra

a hallgató aláírása

Eötvös Loránd Tudományegyetem
Természettudományi Kar

Konszenzusos rangsorok

Szakdolgozat

Borsik Nóra Anna
Matematika BSc
Alkalmazott matematikus szakirány

Témavezető:
Madarasi Péter
ELTE Matematikai Intézet
Operációkutatási Tanszék

Budapest, 2022

Köszönetnyilvánítás

Szeretnék köszönetet mondani témavezetőmnek, Madarasi Péternek a sok segítségért és bátorításért, amellyel hozzájárult a szakdolgozatom létrejöttéhez. Köszönöm a rendszeres konzultációkat és a közös ötleteléseket.

Tartalomjegyzék

1. Bevezetés

A dolgozat témája néhány konszenzusos rangsorolási feladat: Adott versenyzők egy halmaza és bírók egy halmaza. Minden bíró felállít egy szigorú rangsort a versenyzők között, és a feladatunk egy közös rangsor meghatározása, amely valamilyen szempontból jól összesíti az egyes bírók preferenciáit.

Már a 18. században Nicolas de Condorcet és Jean-Charles de Borda francia matematikusok foglalkoztak a témával [10, 14, 17, 26]. Borda javaslata egy pontozásos módszer, amely során minden bíró hozzárendel minden versenyzőhöz annyi pontot, ahányan a sorrendben az adott versenyző mögött végeztek. A különböző bíróktól kapott pontokat összeadjuk, és a kapott pontszámok szerint csökkenő sorrendbe állítjuk a versenyzőket. Ez a módszer minimalizálja a versenyzőkre az átlagos helyezésüktől vett eltérések összegét. Ezzel szemben Condorcet bármely két versenyzőnek az egymáshoz viszonyított helyezését vette alapul. Condorcet kritériuma szerint, ha létezik olyan versenyző, amely páronkénti összehasonlításban minden más versenyzőnél jobb helyen végzett a bírók többségénél, akkor ezen versenyzőnek kell az összesítésben az első helyen végeznie. Az ilyen versenyzőt Condorcet-győztesnek szokás nevezni. Ismert, hogy már három versenyző esetén sem mindig létezik Condorcet-győztes. A korábban leírt Borda módszer nem teljesíti a Condorcet-kritériumot, sőt bizonyítható, hogy semmilyen módszer nem teljesíti, amely az egyes helyezésekhez súlyokat rendel [17].

A továbbiakban mutatunk néhány Condorcet-kritériumot teljesítő módszert egy verseny győztesének meghatározására. Duncan Black javaslata szerint [9], ha létezik Condorcet-győztes, akkor ezen versenyző végezzen az első helyen, ha nem létezik, akkor a Borda-módszer szerinti legmagasabb pontszámú versenyző. Dodgson módszernél azon versenyző nyer, akire a legkevesebb szomszédos csere szükséges a bírók sorrendjeiben, hogy Condorcet-győztessé váljon. A [9] könyvben áttekintik Dodgson munkáját a témával kapcsolatban és a [6] cikkben megmutatják, hogy NP-nehéz kiszámolni egy adott versenyzőre Dodgson-módszer szerint szükséges cserék számát. A Copeland-módszer során egy adott u versenyző 1 pontot kap minden v versenyzőre, akinél jobb helyezésen végzett a bírók többsége szerint, 0 pontot kap azon v versenyzőkre, akik jobb helyezésen végeztek nála a bírók többsége szerint és $\frac{1}{2}$ pontot a többi versenyzőre, akikre ugyanannyi bírónál végzett jobb helyezésen, mint rosszabb helyezésen. Ezután összeadjuk a versenyzők pontjait, és a legtöbb pontot szerző versenyzők közül kerül ki a győztes [12, 28]. A legtöbb csapatsportnál körmérkőzések esetén hasonló módon számolják az egyes csapatok pontjait. A Tideman-módszernél [31] minden (u, v) rendezett versenyzőpárra kiszámoljuk azon bírók számát, akiknél az u versenyző megelőzi a v versenyzőt, és ebből levonjuk azon bírók számát, akiknél a v versenyző megelőzi az u versenyzőt. A kapott értékek szerint csökkenő sorrendbe állítjuk a rendezett versenyzőpárokat, Ezután készítünk egy gráfot, melynek a csúcshalmaza a versenyzők halmaza, és az előbb meghatározott sorrendben minden (u, v) csúcspárra hozzávesszük az uv élt a gráfhoz, ha ezzel nem keletkezik kör. Amikor az összes rendezett páron végigérünk egy teljes irányított gráfot kapunk, amely aciklikus. Ezen gráf egyértelmű topologikus sorrendje szerinti első helyezett a győztes. A itt felsoroltakkal csak néhány példát mutattunk rangsorolási módszerekre, sok más megközelítést is mutatnak például a [31] cikkben.

Kemény János matematikus 1959-ben megfogalmazott egy természetes rangsorolási feladatot [23], amelyre az optimális megoldások teljesítik a Condorcet-kritériumot. A Kemény rangsorolási problémánál olyan közös sorrendet keresünk, amely minimalizálja az egyes bírókhoz képest felcserélt versenyzőpárok számának összegét. Ezt a feladatot a dolgozatban $P_{\sum}$ feladatnak fogjuk nevezni. A [8] cikkben bevezetnek egy hasonló feladatot, melyben a felcserélt versenyzőpárok számának összege helyett az egyes bírókkal szemben felcserélt versenyzők számának maximumát szeretnénk minimalizálni. A dolgozatban ezt a problémát a $P_{\max}$ feladatnak fogjuk nevezni. Szemléletesen olyan sorrendet szeretnénk, amely a legtávolabbi bíróhoz a legközelebb van, azaz egyik bíró preferenciáit sem sérti nagyon. Látható, hogy a $P_{\max}$ feladat már nem teljesíti a Condorcet-kritériumot.

A dolgozatban három fő feladattal fogunk foglalkozni: az előző bekezdésben ismertetett $P_{\sum}$ és $P_{\max}$ feladatokkal és a $P^{v}_{\max}$ feladattal, melyet az 5. Fejezetben vezetünk be. Utóbbinál a bírók helyett a versenyzők szemszögéből definiálunk egy távolságfogalmat, amelynek a maximális értékét szeretnénk minimalizálni.

A dolgozat felépítése a következő: Először megismerkedünk a Kendall-τ távolsággal, amely szemléletesen két bíró esetén azon versenyzők számát adja meg, amelyek egymáshoz viszonyított sorrendjében a két bíró nem ért egyet. A Kendall-τ távolsággal definiáljuk a Kemény János által javasolt $P_{\sum}$ feladatot és a hozzá szorosan kapcsolódó $P_{\max}$ feladatot. A 2. Fejezetben összefoglaljuk az ismert nehézségi eredményeket, többek között látni fogjuk, hogy mindkét probléma már konstans sok bíró esetén is NP-nehéz. A 3. Fejezetben áttekintünk néhány közelítő algoritmust, valamint a $P_{\max}$ problémára adunk egy új, szomszédos versenyzők megcserélésén alapuló algoritmust $k = 3$ bíró esetén. A 4. Fejezetben megmutatjuk a feladatok IP felírását és áttekintjük a legfontosabb FPT algoritmusokat.

Az 5. Fejezetben bevezetünk egy új, $P^{v}_{\max}$-nak nevezett feladatot, amely a versenyzők szemszögéből minimalizálja a közös sorrendtől vett maximális távolságot. Erre a feladatra adunk egy polinomiális algoritmust. A 6. Fejezetben bevezetjük az $(f, g; h)$-tulajdonságú sorrend fogalmát, amely sorrend létezése a $P^{v}_{\max}$ feladat eldöntési verziójának általánosításának tekinthető. Bizonyos feltételek mellett polinomiális algoritmust adunk ilyen sorrend keresésére, és megmutatjuk, hogy a feltételek gyengítésével az eldöntési feladat NP-teljessé válik. Irányított gráfok csúcshalmazának az $(f, g; h)$-tulajdonságú sorrendjeit keresve egy Feedback Arc Set típusú feladatot kapunk. Ennek speciális eseteként megkapjuk annak a karakterizációját, hogy egy irányított gráf mikor áll elő egy aciklikus és egy befenyves uniójaként. Sőt, egy ilyen felbontás polinom időben meghatározható, ha létezik. Megmutatjuk, hogy a feltételeket kissé megváltoztatva már konstans fokszámú gráfok esetén is NP-teljes feladatot kapunk. Zárásként áttekintjük a témával kapcsolatos nyitott kérdéseket.

1.1. Alapfogalmak

A továbbiakban k jelöli a bírók számát, n a versenyzők számát és az egyszerűség kedvéért a versenyzők halmazát megfeleltetjük az első n pozitív egész számnak. Ekkor minden rangsornak megfeleltethető egy $p: \{1, 2, \dots, n\} \to \{1, 2, \dots, n\}$ permutáció, amely minden versenyzőhöz hozzárendeli az adott rangsor szerinti helyezését. Fontos megjegyezni, hogy a p permutáció inverze éppen az adott rangsor szerinti versenyzősorrendet adja meg, például a $p = (3, 2, 4, 1)$ permutáció a $\sigma = p^{-1} = (4, 2, 1, 3)$ versenyzősorrendet fejezi ki. A dolgozatban permutáció alatt a versenyzők permutációját értjük, ha máshogy nem mondjuk. A versenyzők halmazát gyakran V jelöli.

Először definiáljuk két permutációra a Kendall-τ távolságot a [8] cikk alapján:

1.1. Definíció. *A p és q permutációk Kendall-τ távolsága*

$$K(p, q) = |\{(u, v) : p(u) > p(v) \text{ és } q(u) < q(v)\}|.$$

Azaz definíció szerint a Kendall-τ távolság azon (u, v) versenyzőpárok száma, amelyek egymáshoz képest fordított sorrendben szerepelnek a két bírónál. Az ilyen versenyzőpárokra azt mondjuk, hogy a p és a q permutációk között keresztezés van az (u, v) párra.

A következő állítás a Kendall-τ távolságról az 1.3. Következmény miatt lényeges.

1.2. Állítás. *Legyen p és q két permutáció, melyekre $K(p, q) \neq 0$. Ekkor létezik p szerint két szomszédos versenyző, melyekre keresztezés van p és q között.*

Bizonyítás. Mivel $K(p, q) \neq 0$, ezért létezik legalább egy keresztezés p és q között. Vegyünk a p permutáció szerint két versenyzőt, akikre keresztezés van, jelölje őket u és v. Ha u és v szomszédosak p szerint, akkor kész vagyunk, különben van közöttük legalább egy w versenyző, akire biztosan lesz egy (u, w), vagy egy (w, v) keresztezés. Ezt az érvelést a kapott keresztezésre ismételve eljutunk két szomszédos versenyzőhöz, akik között keresztezés van.

1.3. Következmény. *A p és q permutációkra a $K(p, q)$ Kendall-τ távolság megegyezik azon szomszédos cserék minimális számával a p permutáció szerinti sorrendben, melyekkel a p permutációt a q permutációvá alakíthatjuk.*

A továbbiakban definiáljunk a $P_{\max}$ és a $P_{\sum}$ feladatokat a [8] cikk alapján.

1.4. Definíció ($P_{\sum}$ feladat)**.** *A p permutációnak a $P = \{p_1, \dots, p_k\}$ halmaztól vett szumma távolsága*

$$K_{\sum}(P, p) = \sum_{i=1}^{k} K(p_i, p).$$

A $P_{\sum}$ feladatban olyan p permutációt keresünk, amely minimalizálja ezt a távolságot. A szumma távolság szerinti optimum értéket jelölje $OPT_{\sum}(P)$.

1.5. Definíció (*P*$_{\max}$ feladat)**.** *A p permutációnak a $P = \{p_1, \ldots, p_k\}$ halmaztól vett min-max távolsága*

$$K_{\max}(P, p) = \max_{i=1,\ldots,k} K(p_i, p).$$

A $P_{\max}$ feladatban olyan p^* közös permutációt keresünk, amely minimalizálja ezt a távolságot. A min-max távolság szerinti optimum értéket jelölje $OPT_{\max}(P)$.

A $P_{\sum}$ és a $P_{\max}$ feladatok definíciójából következik az alábbi lemma:

1.6. Lemma. *A $P_{\sum}$ és a $P_{\max}$ feladatok szerinti optimum értékekre teljesül, hogy*

$$OPT_{\max} \le OPT_{\sum} \le kOPT_{\max}.$$

Természetes elvárás, hogy ha egy versenyző minden bírónál jobban teljesített egy másik versenyzőnél, akkor a közös sorrendben is jobb helyezésen végezzen. A következő lemma ennek teljesülését mondja ki a feladatok szerinti optimális megoldásokra.

1.7. Lemma. *Ha egy u versenyző minden p_i permutáció szerint megelőzi a v versenyzőt, akkor a $P_{\sum}$, illetve a $P_{\max}$ feladat szerinti minden optimális p^* permutációban teljesül, hogy $p^*(u) < p^*(v)$.*

Bizonyítás. Legyen p egy tetszőleges permutáció, amelyre $p(u) > p(v)$. Cseréljük meg p-ben az u és a v versenyzőt, a kapott permutációt jelölje p'. Megmutatjuk, hogy ekkor $K(p_i, p') < K(p_i, p)$ teljesül minden $p_i \in P$ permutációban. Először belátjuk, hogy nem keletkezhetett az (u, v) pár felcserélésével új keresztezés. Az u előtti és a v utáni versenyzőknek minden más versenyzővel szemben változatlan a sorrendje a csere után, tehát ezen versenyzőkre ugyanannyi keresztezés lesz. Legyen w egy versenyző, amely a p permutációban u és v között végzett. Ekkor ezen versenyzőknek megcserélődött a sorrendje az u és a v versenyzővel szemben. Mivel $p_i(u) < p_i(v)$ minden $p_i \in P$ permutációban, ezért ha $p_i(w) < p_i(u)$, akkor a csere előtt egy (v, w), a csere után egy (u, w) keresztezés lesz. Ha $p_i(v) < p_i(w) < p_i(u)$, akkor a csere előtt egy (u, w) és egy (v, w) keresztezés is volt, a cserével mindkét keresztezés megszűnt. Végül ha $p_i(w) > p_i(v)$, akkor a csere előtt egy (u, w) a csere után egy (v, w) keresztezés lesz. Tehát a cserével nem keletkezhet új keresztezés, azaz minden $p_i \in P$ input permutációra $K(p', p_i) \le K(p, p_i)$ teljesül.

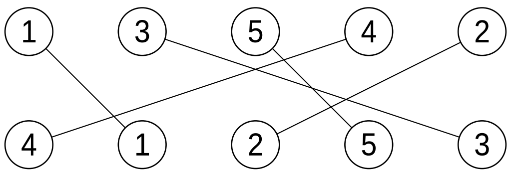
Vegyük észre, hogy az (u, v) párra p és p_i között keresztezés van, de p' és p_i között nincs, tehát Kendall-τ távolság legalább eggyel csökkent minden input permutációra. Ebből következik, hogy a $K_{\sum}$ és $K_{\max}$ távolság is legalább eggyel csökkent, tehát a p permutáció nem lehetett optimális.

Megfigyelés. A $P_{\sum}$ feladatra a Condorcet-kritérium teljesülése hasonló módon bizonyítható.

1.2. Ábrázolás

Két permutáció Kendall-τ távolságát szemléltethetjük páros gráfokkal. Az ábrázoláshoz a [8] cikktől eltérően a permutációk helyett az általuk meghatározott versenyzősorozatokat használjuk a következők szerint: Legyen p_1, p_2 a két permutáció, és tekintsük az inverzeik által meghatározott $\sigma_1 = p_1^{-1}$, $\sigma_2 = p_2^{-1}$ versenyzősorrendeket. Vegyük azt az irányított gráfot, amelyben mindkét csúcsosztály az összes versenyző halmaza és rajzoljuk le a két csúcsosztályt egymást fölé, a felső csúcsosztályt σ_1 az alsó csúcsosztályt σ_2 sorrendben. Minden v versenyzőre menjen egy él a két csúcsosztályban való előfordulása között. Ekkor a $K(p_1, p_2)$ távolság éppen az egyenes élekkel lerajzolt páros gráfban az élek keresztezéseinek száma, mert egy u és egy v versenyzőhöz tartozó élek pontosan akkor keresztezik egymást, ha a két permutációban az u és a v versenyző különböző sorrendben szerepelt.

Például legyen a két permutációnk $p_1 = (1, 5, 2, 4, 3)$ és $p_2 = (2, 3, 5, 1, 4)$, ekkor a távolságuk $K(p_1, p_2) = 6$. A két versenyzősorrend $\sigma_1 = (1, 3, 5, 4, 2)$, $\sigma_2 = (4, 1, 2, 5, 3)$. A távolságot ábrázoló páros gráfot az 1. Ábra szemlélteti.



1. Ábra. A $p_1 = (1, 5, 2, 4, 3)$ és $p_2 = (2, 3, 5, 1, 4)$ permutációkra a Kendall-τ távolság ábrázolása.

1.3. A $P_\sum$ feladat kapcsolata a Feedback Arc Set problémával

A Feedback Arc Set (röviden FAS) problémánál egy adott $D = (V, E)$ irányított gráfon szeretnénk olyan minimális elemszámú $E' \subseteq E$ élhalmazt találni, amelyet elhagyva a maradék $D' = (V, E \setminus E')$ gráf aciklikus. Ismert, hogy általános gráfokon a FAS probléma NP-nehéz [22], valamint APX-nehéz, mivel a minimális lefogó csúcshalmaz probléma visszavezethető rá L-redukcióval [21]. A visszavezetésből következik, hogy általános gráfokon a FAS problémára nem létezhet 1.36-közelítő algoritmus, ha P $\neq$ NP [16]. Ha igaz a Unique Games Conjecture (UGC) [25], akkor semmilyen c konstansra nem létezhet c-közelítő algoritmus [20]. A legjobb ismert közelítő hányados $O(\log n \log \log n)$ [18, 30]. A FAS probléma turnamenteken is NP-nehéz, azonban turnamenteken létezik rá PTAS [24]. A FAS problémát általánosíthatjuk élsúlyozott irányított gráfokra. A súlyozott FAS problémánál egy adott $D = (V, E)$ élsúlyozott irányított gráfon szeretnénk olyan minimális súlyösszegű $E' \subseteq E$ élhalmazt találni, amelyet elhagyva a maradék $D' = (V, E \setminus E')$ gráf aciklikus.

Megmutatjuk a $P_\sum$ feladat visszavezetését a súlyozott FAS probléma oda-vissza irányított teljes gráfokon vett speciális esetére, amelyre létezik PTAS [24]. Adott P permutációhalmazra jelölje minden u, v versenyzőpárra $A_P(u, v)$ azon $p \in P$ permutációk számát, melyekben az u versenyző megelőzi a v versenyzőt. Világos, hogy $A_P(u, v) + A_P(v, u) = k$, ahol k jelöli a bírók számát.

Adott P permutációhalmazra definiáljuk a büntetési gráf fogalmát, amelyet számos cikkben felhasználnak [2, 13, 24].

1.8. Definíció. *Legyen a büntetési gráf egy oda-vissza irányított teljes gráf, amelynek csúcsai a versenyzőknek felelnek meg és minden u, v versenyzőpárra az uv él súlya legyen $w(uv) = \frac{A_P(u,v)}{k}$, azaz azon bírók aránya a P halmazban, melyekben az u versenyző megelőzi a v versenyzőt.*

Megmutatjuk, hogy a D büntetési gráf minimális súlyösszegű FAS-jei megfelelnek a $P_{\sum}$ feladat optimális megoldásainak. Vegyük észre, hogy a p permutációnak megfeleltethető a permutáció által meghatározott sorrend szerinti visszaélek $E' \subseteq E$ halmaza. Vegyük észre, hogy a p permutációnak P halmaztól vett távolsága megegyezik az E' élhalmaz súlyösszegének k-szorosával, azaz

$$K_{\sum}(P,p) = \sum_{p(u)>p(v)} A_P(u,v) = k \sum_{e \in E'} w(e),$$

ahol az első egyenlőség a szumma távolság definíciójából következik, a második egyenlőség pedig a D gráf és az E' élhalmaz definíciójából. Tehát egy p permutáció pontosan akkor optimális a $P_{\sum}$ feladatra, ha az általa meghatározott E' élhalmaz optimális a súlyozott FAS feladatra. A fordított irány igazolásához vegyük észre, hogy tetszőleges $E' \subseteq E$ FAS-ra, a $D' = (V, E \setminus E')$ aciklikus gráf bármely topologikus sorrendje által meghatározott p permutáció $K_{\sum}(P,p)$ költsége megegyezik az E' súlyösszegének k-szorosával.

Látható, hogy az így definiált büntetési gráfban minden (u,v) csúcspárra az uv és a vu élek súlyának összege 1, ezért alkalmazható rá a [24] cikkben adott PTAS.

2. Nehézségi eredmények

Ebben a fejezetben áttekintjük a $P_{\sum}$ és $P_{\max}$ feladatra az ismert nehézségi eredményeket. Mindkét problémáról ismert, hogy már konstans sok bíró esetén is NP-nehezek [8]. Megmutatjuk, hogy a $P_{\sum}$ feladat visszavezetése a $P_{\max}$ feladatra approximáció-tartó.

2.1. A $P_{\sum}$ feladat

A következő tételt a $P_{\sum}$ feladat nehézségéről az [5] és a [8] cikkek alapján mondjuk ki.

2.1. Tétel. *A $P_{\sum}$ feladat $k \geq 4$ páros, és $k \geq 7$ páratlan számú bíróra NP-nehéz.*

A [8] cikkben kijavítják a [17] cikkben mutatott konstrukciót, amely bizonyítja, hogy a $P_{\sum}$ feladat $k \geq 4$ páros számú bírók esetén NP-nehéz. A bizonyítás során először visszavezetik a Feedback Arc Set problémát a $P_{\sum}$ feladatra $k = 4$ bíró esetén, majd megmutatják, hogy adott P halmazhoz egy permutációt és a fordítottját hozzávéve ugyanazok az optimális permutációk. Az [5] cikkben megmutatják, hogy a probléma NP-nehéz $k \geq 7$ esetén páratlan sok bíróra is. Később látni fogjuk, hogy a $P_{\sum}$ feladat $k = 2$ bíró esetén megoldható. A $k = 3$ és $k = 5$ bíró esetének nehézsége tudomásunk szerint nyitott kérdés.

2.2. A $P_{\max}$ feladat

Megmutatjuk, hogy a $P_{\max}$ feladat legalább 4 bíró esetén NP-nehéz. A feladat $k = 2$ bíró esetén megoldható, erre később mutatunk egy algoritmust a 3.2.2. Fejezetben. Tudomásunk szerint $k = 3$ bíró esetének nehézsége nyitott kérdés.

A következő tételt és bizonyítását a [8] cikk alapján mondjuk ki.

2.2. Tétel. *A $P_{\max}$ feladat NP-nehéz $k \geq 4$ bíró esetén.*

Bizonyítás. Visszavezetjük rá a $P_{\sum}$ feladatot $k = 4$ bíró esetén. A permutációhalmazt, amelyre meg szeretnénk oldani a $P_{\sum}$ feladatot jelölje $P = \{p_1, p_2, p_3, p_4\}$. Definiáljuk a $Q = \{q_1, q_2, q_3, q_4\}$ halmazt, ahol a q_i permutációk az alábbiak:

$$
\begin{aligned}
q_1 &= p_1[1, \ldots, n]\, p_2[n+1, \ldots, 2n]\, p_3[2n+1, \ldots, 3n]\, p_4[3n+1, \ldots, 4n] \\
q_2 &= p_2[1, \ldots, n]\, p_3[n+1, \ldots, 2n]\, p_4[2n+1, \ldots, 3n]\, p_1[3n+1, \ldots, 4n] \\
q_3 &= p_3[1, \ldots, n]\, p_4[n+1, \ldots, 2n]\, p_1[2n+1, \ldots, 3n]\, p_2[3n+1, \ldots, 4n] \\
q_4 &= p_4[1, \ldots, n]\, p_1[n+1, \ldots, 2n]\, p_2[2n+1, \ldots, 3n]\, p_3[3n+1, \ldots, 4n],
\end{aligned}
$$

ahol a $p_i[a, \ldots, b]$ azt jelöli, hogy az $a, \ldots, b$ versenyzők a $a, \ldots, b$ helyezéseken végeztek, a p_i permutáció szerinti sorrendben. Szemléletesen a versenyzőket 4 darab n elemű blokkra tudjuk osztani, hogy a j-edik blokkban a versenyzők minden q_i permutáció szerint a $((j-1)n + 1)$-edik és az jn-edik helyezés között végeztek.

Jelölje q^* erre a Q halmazra a $P_{\sum}$ feladat optimális megoldását, az optimum értékét pedig $OPT_{\sum}(Q)$. Ekkor az 1.7. Lemmából következik, hogy a q^* megoldás is 4 diszjunkt blokkból áll, azaz

$$q^* = p_1^*[1, \ldots, n]\, p_2^*[n+1, \ldots, 2n]\, p_3^*[2n+1, \ldots, 3n]\, p_4^*[3n+1, \ldots, 4n].$$

A Q halmazban minden blokkban minden p_i permutációt pontosan egy q_i permutációban alkalmaztunk, ezért az $OPT_{\sum}(Q)$ értékre teljesül, hogy

$$OPT_{\sum}(Q) = K_{\sum}(Q, q^*) = \sum_{i=1}^{4} K_{\sum}(P, p_i^*).$$

Ebből látszik, hogy p_1^*, p_2^*, p_3^* és p_4^* is optimális megoldás a P halmazon a $P_{\sum}$ feladatra, ezért következik az alábbi egyenlőség:

$$OPT_{\sum}(Q) = 4 OPT_{\sum}(P). \tag{1}$$

Vegyük észre, hogy ha minden blokkra ugyanazt a p_i permutációt alkalmazzuk, például vesszük a $q^0 = p_1[1, \dots, n]\, p_1[n+1, \dots, 2n]\, p_1[2n+1, \dots, 3n]\, p_1[3n+1, \dots, 4n]$ permutációt, akkor az így kapott permutáció is optimális megoldás a Q halmazon a $P_{\sum}$ feladatra. A q^0 permutáció definíciójából, valamint P és a Q halmaz kapcsolatából látható, hogy

$$OPT_{\sum}(P) = K_{\sum}(P, p_1) = K_{\max}(Q, q^0). \tag{2}$$

A két számozott egyenlőségből következik, hogy $OPT_{\sum}(Q) = 4K_{\max}(Q, q^0)$. Ebből az 1.6. Lemma miatt következik, hogy q^0 optimális a Q halmazon a $P_{\max}$ feladatra, és a Q halmazon a $P_{\max}$ feladat minden megoldása egyben a $P_{\sum}$ feladatra is optimális. Tehát $OPT_{\sum}(P) = OPT_{\max}(Q)$, és az optimális megoldás a P halmazon a $P_{\sum}$ feladatra kiszámolható, ha a Q halmazra, megoldjuk a $P_{\max}$ feladatot, és a megoldásnak vesszük bármelyik blokkját. Ezzel kész a bizonyítás $k = 4$ bíróra, a $k > 4$ esetnél megtehetjük, hogy a q_1 permutációt $(k-4)$-szer duplikáljuk a Q halmazban, ezzel az $OPT_{\max}(Q)$ érték nem változik, és a 4 bíróra vett $P_{\sum}$ feladatot visszavezettük a $P_{\max}$ feladatra $k > 4$ bíró esetén.

Megfigyelés. Ha $k > 4$ bíróra a $P = \{p_1, p_2, \dots, p_k\}$ $P_{\sum}$ feladathoz, a $Q = \{q_1, q_2 \dots q_k\}$ permutációkat az alábbi módon definiáljuk:

$$\begin{aligned}
q_1 &= p_1[1, \dots, n]\, p_2[n+1, \dots, 2n] \dots p_k[(k-1)n+1, \dots, kn] \\
q_2 &= p_2[1, \dots, n]\, p_3[n+1, \dots, 2n] \dots p_1[(k-1)n+1, \dots, kn] \\
&\vdots \\
q_k &= p_k[1, \dots, n]\, p_1[n+1, \dots, 2n] \dots p_{k-1}[(k-1)n+1, \dots, kn],
\end{aligned}$$

akkor ugyanúgy működik a bizonyítás, mint $k = 4$ bíróra. Erre szükség lesz a következő bizonyításban.

A továbbiakban megmutatjuk, hogy a fenti visszavezetés approximáció-tartó.

2.3. Állítás. *A $P_{\sum}$ feladat visszavezetése a $P_{\max}$ feladatra approximáció-tartó, azaz ha ismerünk egy α-közelítő algoritmust a $P_{\max}$ feladatra, akkor ezt az algoritmust a visszavezetés szerinti permutációkon alkalmazva a $P_{\sum}$ feladatra is α-közelítő algoritmust kapunk.*

Bizonyítás. Jelölje $P = \{p_1, p_2, \dots, p_k\}$ a permutációhalmazt, amelyre meg szeretnénk oldani a $P_{\sum}$ feladatot. Definiáljuk a Q halmazt a visszavezetés szerint, de 4 bíró helyett általánosan k bíróra. Vegyünk egy α-közelítő megoldás a Q halmazon a $P_{\max}$ feladatra. Ez a megoldás nem feltétlen áll diszjunkt blokkokból, de a visszavezetés bizonyításánál láttuk, hogy blokkok szerinti rendezéssel a keresztezések száma a $P_{\sum}$ feladatra nézve nem nőhet. Ugyanez igaz a $P_{\max}$ feladatra is, mert a rendezés során nem jön létre új keresztezés semelyik q_i permutációra nézve. Tehát az α-közelítő megoldás blokkok szerinti rendezettje is α-közelítő. Legyen a blokkok szerint rendezett permutáció

$$q^* = p_1[1, \dots, n]\, p_2[n+1, \dots, 2n] \dots p_k[(k-1)n+1, \dots, kn].$$

Ekkor

$$K_{\sum}(Q, q^*) \le kK_{\max}(Q, q^*) \le k\alpha OPT_{\max}(Q) = k\alpha OPT_{\sum}(P), \tag{3}$$

ahol az első egyenlőtlenség az 1.6. Lemma állítása, a második egyenlőtlenség abból következik, hogy a q permutáció α-közelítő a $P_{\max}$ feladatra, az egyenlőség pedig a visszavezetésből adódik.

Jelölje p azt a permutációt $p_1, p_2, \ldots, p_k$ közül, amelyre a $K_{\sum}(P, p_i)$ távolság minimális. Ekkor teljesül a következő becslés:

$$K_{\sum}(P, p\ ) \quad \frac{1}{k} K_{\sum}(Q, q\ ) \quad \alpha OPT_{\sum}(P),$$

ahol az első egyenlőtlenség a visszavezetésből és p választásából következik, a második egyenlőtlenség pedig megegyezik a (3) egyenlőtlenséggel, csak minkét oldalt osztottuk k-val. Tehát a p permutáció α-közelítő.

2.4. Következmény. *A 2.3. Állításból következik, hogy ha létezik egy PTAS a $P_{\max}$ feladatra, akkor ebből a visszavezetéssel kapunk egy PTAS-t a $P_{\sum}$ feladatra.*

3. Közelítő algoritmusok

Ebben a fejezetben áttekintünk néhány közelítő algoritmust először a $P_{\sum}$, majd a $P_{\max}$ feladatra. Mindkét problémára mutatunk olyan algoritmust, amely $k = 2$ bíró esetén optimális megoldást ad.

3.1. A $P_{\sum}$ feladat

A következő fejezetben áttekintünk néhány közelítő algoritmust a $P_{\sum}$ feladatra.

3.1.1. A legjobb input permutáció

A PickAPerm$_{\sum}$ algoritmus során veszünk egy p_i 2 P permutációt a $P = f p_1, \ldots, p_k g$ permutációk közül, melyre a $K_{\sum}(P, p_i)$ érték minimális. A [8] cikk alapján megmutatjuk, hogy ez az algoritmus $\left(2 \quad \frac{2}{k}\right)$-közelítő a $P_{\sum}$ feladatra.

3.1. Tétel. *A PickAPerm$_{\sum}$ algoritmus $\left(2 \quad \frac{2}{k}\right)$-közelítő a $P_{\sum}$ feladatra.*

Bizonyítás. A tételt először páratlan k bírók száma esetén fogjuk bizonyítani, felhasználva az 1.3. Fejezetben definiált büntetési gráfot, amelyre láttuk, hogy minden FAS élhalmaznak megfelel egy permutáció, melynek a $P_{\sum}$ feladat szerinti költsége pontosan k-szorosa a FAS súlyösszegének. A büntetési gráfban minden (u, v) csúcspárra teljesül a $w(uv) + w(vu) = 1$ egyenlőség. Mivel k páratlan, ezért egyik él súlya sem pontosan $\frac{1}{2}$, tehát minden (u, v) csúcspárra az uv és a vu élek közül a nagyobb súlyú él súlya nagyobb mint $\frac{1}{2}$, a másik él súlya kisebb mint $\frac{1}{2}$. Vegyük észre, hogy az összes kisebb súlyú él súlyának összege alsó becslés az optimális FAS súlyösszegére, az alsó becslést jelölje L.

Jelölje m_a az $\frac{a}{k}$ és $\frac{k-a}{k}$ súlyú oda-vissza élpárok számát. Ezzel a jelöléssel

$$L = \sum_{a=\lceil \frac{k}{2} \rceil}^{k} m_a \frac{k-a}{k}.$$

A büntetési gráf definíciója miatt egy $\frac{a}{k}$ súlyú él pontosan $(k-a)$ input permutáció szerinti sorrendben visszaél. Ekkor az összes input permutációra a nagyobb súlyú visszaélek számának összege $\sum_{a=\lceil \frac{k}{2} \rceil}^{k} m_a(k-a)$. Tehát létezik legalább egy olyan p_i input permutáció, hogy a p_i szerinti nagyobb súlyú visszaélek száma legfeljebb

$$\frac{1}{k} \sum_{a=\lceil \frac{k}{2} \rceil}^{k} m_a(k-a) = \sum_{a=\lceil \frac{k}{2} \rceil}^{k} m_a \frac{k-a}{k} = L. \tag{4}$$

Jelölje r_a a p_i permutáció szerinti $\frac{a}{k}$ súlyú visszaélek számát. Ekkor a p_i permutáció szerinti visszaélek súlyösszege felírható úgy, hogy az alsó becslés értékéhez hozzáadjuk a nagyobb súlyú visszaélekre keletkező plusz súlyokat, azaz

$$L + \sum_{a=\lceil \frac{k}{2} \rceil}^{k} r_a \left(\frac{a}{k} - \frac{k-a}{k} \right) = L + \sum_{a=\lceil \frac{k}{2} \rceil}^{k-1} r_a \left(\frac{a}{k} - \frac{k-a}{k} \right) \le$$

$$\le L + \sum_{a=\lceil \frac{k}{2} \rceil}^{k-1} r_a \frac{k-2}{k} \le \left(2 - \frac{2}{k}\right) L,$$

ahol az egyenlőség azért igaz, mert $a = k$ esetén $r_a = 0$, mivel semelyik input permutáció szerint sem fordulhat meg 1 súlyú él. Az első egyenlőtlenség abból következik, hogy az $\left(\frac{a}{k} - \frac{k-a}{k}\right)$ különbség akkor maximális, amikor $a = k - 1$ és ekkor az értéke $\frac{k-2}{k}$. Az utolsó egyenlőtlenség abból következik, hogy a $\sum_{a=\lceil \frac{k}{2} \rceil}^{k-1} r_a$ összeg megegyezik a p_i szerinti nagyobb súlyú visszaélek számával amelyre a (4) becslésnél láttuk, hogy legfeljebb L. Tehát a p_i permutáció szerinti FAS súlyösszege $\left(2 - \frac{2}{k}\right)$-közelítő a büntetési gráfon a súlyozott FAS feladatra, ebből következik, hogy a p_i permutáció $\left(2 - \frac{2}{k}\right)$-közelítő a $P_{\sum}$ feladatra.

A bizonyítás páros k bírók száma esetén ugyanígy működik, csak az $\frac{1}{2}$ súlyú élek esetében az uv és vu élek közül az egyikét kisebb súlyú éleknél, a másikat a nagyobb súlyú éleknél kell számolnunk.

3.2. Következmény. *A $P_{\sum}$ feladatra $k = 2$ bíró esetén a PickAPerm$_{\sum}$ algoritmus optimális. A $P = \{p_1, p_2\}$ inputra p_1 és p_2 is optimális megoldás és $OPT_{\sum} = K(p_1, p_2)$, mivel $K_{\sum}(P, p_1) = K(p_1, p_2) = K_{\sum}(P, p_2)$.*

3.1.2. Spearman footrule

A $P_{\sum}$ és a $P_{\max}$ feladatokat a Kendall-τ távolság szerint definiáltuk. A fejezetben bevezetünk egy másik ismert távolságfogalmat, amely szerinti optimális megoldás 2-közelítő a $P_{\sum}$ feladatra [15, 17].

3.3. Definíció. *A p és q permutációk Spearman footrule távolsága*

$$F(p,q) = \sum_{v \in V} |p(v) - q(v)|.$$

Egy p permutációnak a $P = \{p_1, \ldots, p_k\}$ halmaztól vett $F_{\sum}$ távolsága

$$F_{\sum}(P,p) = \sum_{i=1}^{k} F(p_i, p).$$

A következő tételt Kendall-τ és Spearman footrule távolságok kapcsolatáról a [15] cikk alapján mondjuk ki.

3.4. Tétel. *Tetszőleges p és q permutációkra teljesül, hogy $K(p,q) \leq F(p,q) \leq 2K(p,q)$.*

Bizonyítás. Először a jobb oldali egyenlőtlenséget bizonyítjuk. Az 1.3. Következmény szerint a Kendall-τ távolság megegyezik azon p szerinti szomszédos cserék minimális számával, melyekkel a p permutációt átalakíthatjuk a q permutációvá. Egy szomszédos csere során az $F(p,q)$ távolság vagy kettővel csökken, vagy nem változik, mivel ha u és v között keresztezés volt, akkor a megcserélésükkel legalább az egyik versenyző közelebb kerül a q permutáció szerinti helyezéséhez. Ha a másik versenyző is közelebb kerül a q szerinti helyezéséhez, akkor az $F(p,q)$ távolság kettővel csökken, ha a másik távolabb kerül, akkor az $F(p,q)$ távolság nem változik. Ebből következik, hogy $F(p,q) \leq 2K(p,q)$.

A bal oldali egyenlőtlenség igazolásához jelölje a $p(u) > p(v)$ és $q(u) < q(v)$ keresztezést (u,v). Az (u,v) keresztezést 1. típusúnak nevezzük, ha $p(u) \geq q(v)$ és 2. típusúnak, ha $p(u) \leq q(v)$. Így minden keresztezés legalább az egyik kategóriába tartozik, de lehet olyan amelyik mindkettőbe. Most adunk egy felső becslést a különböző típusú keresztezések számára.

Az 1. típusú keresztezések esetén $q(u) < q(v) \leq p(u)$, ezért egy adott u versenyzőre az (u,x) 1. típusú keresztezések száma legfeljebb $p(u) - q(u)$.

A 2. típusú keresztezések esetén $p(v) < p(u) \leq q(v)$, ezért egy adott v versenyzőre az (x,v) 2. típusú keresztezések száma legfeljebb $q(v) - p(v)$.

Ezekből következik az alábbi egyenlőtlenség:

$$K(p,q) \leq \sum_{p(u) \geq q(u)} (p(u) - q(u)) + \sum_{p(v) \leq q(v)} (q(v) - p(v)) = \sum_{v \in V} |p(v) - q(v)| = F(p,q),$$

ahol az első egyenlőtlenség azért igaz, mert a Kendall-τ távolság megegyezik az összes keresztezés számával és legfeljebb annyi keresztezés van ahány 1. típusú és 2. típusú keresztezés összesen. Az első egyenlőség az abszolút érték definíciójából, a második egyenlőség pedig a Spearman footrule távolság definíciójából következik.

3.5. Következmény. *Adott p permutációnak a P halmaztól vett távolságára teljesül, hogy $K_{\sum}(p,q) \leq F_{\sum}(P,p) \leq 2K_{\sum}(P,p)$. Ebből következik, hogy az $F_{\sum}$ távolságot minimalizáló p^* permutációra a $K_{\sum}(P,p^*)$ távolság legfeljebb kétszerese az $OPT_{\sum}$ értéknek.*

A [17] cikk alapján megmutatjuk, hogy adott $P = \{p_1, \dots, p_k\}$ permutációhalmazon polinom időben tudunk találni egy közös permutációt, amely minimalizálja az $F_{\sum}$ távolságot. Legyen $G = (H \cup V, E)$ egy teljes páros gráf, melynek H csúcsosztálya a helyezéseknek felel meg, V csúcsosztálya pedig a versenyzőknek. Minden $h \in H$, $v \in V$ csúcspárra a hv él súlya legyen $\sum_{i=1}^{k} |p_i(v) - h|$. Vegyük észre, hogy a G gráf teljes párosításai egyértelműen megfeleltethetőek a permutációknak. Egy p permutációnak akkor felel meg az M teljes párosítás, ha minden $hv \in M$ élre $p(v) = h$ teljesül. Ha egy p permutációnak megfelel egy M teljes párosítás, akkor az $F_{\sum}(P, p)$ távolság éppen az M párosítás súlya, mivel a következő egyenlőség teljesül:

$$F_{\sum}(P,p) = \sum_{i=1}^{k} F(p_i, p) = \sum_{i=1}^{k} \sum_{v \in V} |p_i(v) - p(v)| = \sum_{v \in V} \sum_{i=1}^{k} |p_i(v) - p(v)| = \sum_{hv \in M} \sum_{i=1}^{k} |p_i(v) - h|,$$

ahol az első és a második egyenlőség a 3.3. Definícióból következik, a harmadik egyenlőség egyszerű algebrai átalakítás, a negyedik egyenlőségben pedig kihasználtuk, hogy p permutációnak megfelelt az M párosítás, ezért $p(v) = h$. Az egyenlőségek jobb oldalán az M párosítás súlyát kaptuk. Ebből következik, hogy az $F_{\sum}$ távolságot minimalizáló permutációt meg tudjuk keresni a G teljes páros gráfban egy minimális súlyú teljes párosítás keresésével. A 3.5. Következmény miatt az így kapott permutáció 2-közelítő a $P_{\sum}$ feladatra.

Megfigyelés. Minden T távolság szerint, amely minden versenyző-helyezés párhoz súlyt rendel, tudunk a $T_{\sum}(P, p) = \sum_{i=1}^{k} T(p_i, p)$ távolságot minimalizáló közös permutációt keresni polinom időben. A feladat az előbbi konstrukcióhoz hasonlóan visszavezethető egy teljes páros gráfban való minimális súlyú teljes párosítás keresésére.

3.1.3. Borda-módszer

A Borda-módszer egy gyakran használt pontozásos rangsorolási módszer [10, 17, 26]. Ha a versenyzők száma n, akkor minden bíró felállít egy versenyzősorrendet, és az első helyezetthez $(n - 1)$ pontot rendel, a másodikhoz $(n - 2)$ pontot és így tovább, az utolsó versenyzőhöz 0 pontot. Ezek után minden versenyzőre összeadjuk a k bíró által adott pontokat és a versenyzőket a kapott pontszámok szerint csökkenő sorrendbe állítjuk. Előfordulhat, hogy néhány versenyzőnek megegyezik a pontszáma, ilyen esetekben az azonos pontszámú versenyzőket egymáshoz képest tetszőleges sorrendben rögzítjük.

A Borda-módszert megfogalmazhatjuk úgy is, hogy minden v versenyzőre kiszámoljuk a $P = \{p_1, \dots, p_k\}$ permutációk szerinti átlagos helyezését, azaz a $\frac{\sum_{i=1}^{k} p_i(v)}{k}$ értéket, és az átlagos helyezésük szerinti növekvő sorrendbe rendezzük a versenyzőket (azonos átlagos helyezés esetén tetszőleges sorrendbe). Vegyük észre, hogy a két definíció ekvivalens.

A második definíciót felhasználva megmutatjuk a Borda módszer kapcsolatát az 1.3. Fejezetben definiált büntetési gráffal. A büntetési gráfban a v csúcs súlyozott befokát jelölje

$\delta^w(v)$. Ekkor

$$\delta^w(v) = \sum_{u \in V, u \neq v} \frac{A_P(u,v)}{k} = \frac{\sum_{i=1}^{k}(p_i(v) - 1)}{k} = \frac{\sum_{i=1}^{k} p_i(v)}{k} - 1,$$

mivel egy adott v csúcsra a $\sum_{u \in V, u \neq v} A_P(u,v)$ összeg megegyezik azzal, hogy összeadjuk a v versenyzőt megelőző versenyzők számát az egyes permutációk szerint. Egy p_i permutációban a v versenyzőt megelőző versenyzők száma éppen $p_i(v) - 1$. A fenti egyenletből következik, hogy adott P permutációhalmazra a Borda-módszer szerinti sorrend megegyezik az 1.3. Fejezetben bevezetett büntetési gráf csúcsainak a súlyozott befokok szerinti növekvő sorrendjével.

A következő tételt és bizonyítását a [13] cikk alapján mondjuk ki.

3.6. Tétel. *Adott egy $D = (V, E)$ élsúlyozott, oda-vissza irányított teljes gráf, az élein egy $w : E \to \mathbb{R}_+$ súlyozással, melyre minden (u, v) párra $w(uv) + w(vu) = 1$. Ekkor a csúcsokat a súlyozott befokok szerint növekvő sorrendbe rendezve a sorrend szerint balra menő élek $E' \subseteq E$ halmaza 5-közelítő megoldás a súlyozott FAS problémára a D gráfon.*

Bizonyítás. A bizonyításhoz először bevezetjük a következő jelöléseket: A D gráf egy v csúcsának a súlyozott kifokát jelölje $\delta^w(v)$, a súlyozott befokát pedig $\varrho^w(v)$. A D gráfon értelmezett FAS feladatra az egyik optimális élhalmazhoz tartozó permutációt jelölje p^*, a csúcsok befok szerint növekvő sorrendjét megadó permutációt pedig p_B. A csúcsok tetszőleges p permutációjára E_p jelöli a p szerint balra menő élek halmazát és $w(E_p)$ az élek súlyösszegét. A tételt a következő becslés igazolásával látjuk be:

$$\begin{aligned} 4w(E_{p^*}) &\geq 2\sum_{v \in V} |p^*(v) - 1 - \varrho^w(v)| \geq \sum_{v \in V} |p^*(v) - 1 - \varrho^w(v)| + \sum_{v \in V} |p_B(v) - 1 - \varrho^w(v)| \\ &\geq \sum_{v \in V} |p^*(v) - p_B(v)| = F(p^*, p_B) \geq K(p^*, p_B) \geq w(E_{p_B}) - w(E_{p^*}). \quad (5) \end{aligned}$$

Az első egyenlőtlenség igazolásához jelölje a $\overleftarrow{\delta}^w(v)$ a v csúcs p^* permutáció szerinti bal súlyozott kifokát, azaz a v csúcsot a p^* permutáció szerint megelőző csúcsokra megszorított súlyozott kifokát. Hasonlóan $\vec{\delta}^w(v)$ jelöli a jobb súlyozott kifokot, $\vec{\varrho}^w(v)$ a bal súlyozott befokot és $\overleftarrow{\varrho}^w(v)$ a jobb súlyozott befokot. Ezekkel a jelölésekkel minden v csúcsra teljesül a következő egyenlőség:

$$\varrho^w(v) = \vec{\varrho}^w(v) + \overleftarrow{\varrho}^w(v), \quad (6)$$

azaz egy v csúcs súlyozott befoka megegyezik a bal súlyozott befokának és a jobb súlyozott befokának összegével. Szükségünk lesz a következő egyenlőségre is:

$$p^*(v) - 1 = \vec{\varrho}^w(v) + \overleftarrow{\delta}^w(v). \quad (7)$$

Mivel $w(uv) + w(vu) = 1$, ezért egy v csúcsra a v és az őt megelőző csúcshalmaz közötti élek súlyainak az összege megegyezik a v csúcsot megelőző csúcsok számával. Ezeket felhasználva bizonyítható a következő egyenlőtlenség:

$$2w(E_{p}\ ) = \sum_{v \in V} \overleftarrow{\delta}^{w}(v) + \overleftarrow{\varrho}^{w}(v) \quad \sum_{v \in V} |\overleftarrow{\delta}^{w}(v) \quad \overleftarrow{\varrho}^{w}(v)| = \sum_{v \in V} |p\ \ (v) \quad 1 \quad \varrho^{w}(v)|,$$

ahol az első egyenlőség azért igaz, mert minden balra élt számolunk a kezdőpontjának a kifokánál és a végpontjának a befokánál, az egyenlőtlenség azért igaz, mert $\overleftarrow{\delta}^{w}(v)$ és $\overleftarrow{\varrho}^{w}(v)$ is nemnegatív és az utolsó egyenlőség pedig a (6) és a (7) egyenlőségek miatt teljesül. Tehát az (5) becslés első egyenlőtlensége teljesül.

Az (5) becslésben a második egyenlőtlenség abból következik, hogy a p_B permutáció minimalizálja a $\sum_{v \in V} |p(v) \quad 1 \quad \varrho^{w}(v)|$ mennyiséget. A harmadik egyenlőtlenség a háromszög-egyenlőtlenségből következik, az egyenlőség a Spearman footrule távolság definíciója miatt igaz, a negyedik egyenlőtlenség pedig az előző fejezetben kimondott 3.4. Tétel állítása.

Az (5) becslésben az ötödik egyenlőtlenség bizonyításához jelölje $E_{p\ \ n p_B}$ azon visszaélek halmazát, amelyek csak a $p\ $ permutáció szerint visszaélek. Ezzel a jelöléssel teljesül a következő egyenlőtlenség:

$$K(p\ \ , p_B) = w(E_{p\ \ n p_B}) + w(E_{p_B n p\ \ }) \quad w(E_{p_B}) \quad w(E_{p\ \ }),$$

ahol az első egyenlőség azért igaz, mert ha egy (u, v) csúcspárra nincs keresztezés a két permutáció között, akkor az uv él vagy mindkét permutáció szerint előreél, vagy mindkét permutáció szerint visszaél. Tehát a $w(E_{p\ \ n p_B}) + w(E_{p_B n p\ \ })$ összegben csak azon uv élek súlyát számoljuk, melyekre az u és v csúcsokra keresztezés van a két permutáció között. Ezen (u, v) pároknál az egyik permutáció szerint uv visszaél, a másik szerint pedig vu, tehát minden keresztezésre $w(uv) + w(vu) = 1$ értéket számolunk az összegben. Az egyenlőtlenség az $E_{p\ \ n p_B}$ és az $E_{p_B n p\ \ }$ definíciójából következik.

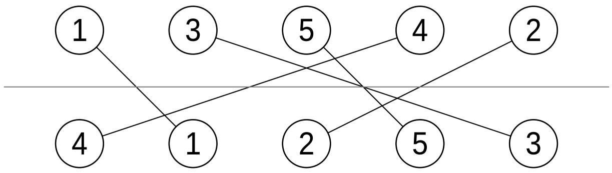
3.7. Következmény. *A Borda-módszer 5-közelítő megoldást ad a $P_{\sum}$ feladatra, mivel a büntetési gráf teljesíti az előző tétel feltételeit és az 1.3. Fejezetben láttuk, hogy a büntetési gráfon a súlyozott FAS probléma ekvivalens a $P_{\max}$ feladattal.*

A [13] cikkben adnak egy gráfcsaládot, amely bizonyítja, hogy gráfokon ez a becslés éles, azonban nyitott kérdés, hogy ez igaz-e a $P_{\sum}$ feladatra is. A továbbiakban teszünk néhány megfigyelést a Borda-módszerről $k = 2$ bíró esetén.

3.8. Állítás. *A Borda-módszer $k = 2$ bíró esetén optimális megoldást ad a $P_{\sum}$ feladatra.*

Bizonyítás. Jelölje p_B a Borda-módszerből kapott közös permutációt. Vegyük észre, hogy ha az u versenyző megelőzi a v versenyzőt p_1 szerint és p_2 szerint is, akkor p_B szerint is, mivel u átlagos helyezése biztosan kisebb, mint v átlagos helyezése. Tehát p_B szerint csak olyan (u, v) versenyzőpárokra lehet keresztezés p_1 vagy p_2 felé, melyeknek különböző a sorrendje a két permutáció szerint. Az ilyen (u, v) párokra p_B szerint pontosan egy keresztezés lesz. Ebből következik, hogy $K_{\sum}(P, p_B) = K(p_1, p_2)$. Korábban a 3.1.1. Fejezetben láttuk, hogy a $P = \{p_1, p_2\}$ permutációhalmazra a $K(p_1, p_2)$ érték optimális a $P_{\sum}$ feladatra, tehát a p_B permutáció is optimális a $P_{\sum}$ feladatra.

Megfigyelés. A Borda-módszer szerinti p_B permutáció $k = 2$ bíró esetén az 1.2. Fejezetben mutatott ábrázolás segítségével könnyen szemléltethető. Ábrázoljuk a két permutációt és húzzunk egy vízszintes vonalat a két permutáció között félúton. A Borda-módszer megoldása megegyezik azzal a sorrenddel, amilyen sorrendben az egyes versenyzőkhöz tartozó élek metszik a vízszintes vonalat.



2. Ábra. A $p_1 = (1, 5, 2, 4, 3)$ és $p_2 = (2, 3, 5, 1, 4)$ permutációkra a Borda-módszer ábrázolása.

Például a $p_1 = (1, 5, 2, 4, 3)$ és $p_2 = (2, 3, 5, 1, 4)$ permutációkra a 2. ábráról leolvasható, hogy a Borda-módszer szerint két sorrend lehetséges mivel a 3 és 5 versenyzőknek azonos az átlagos helyezése (a vízszintes vonalat egy pontban metszi a két él). Ekkor a két Borda-módszer szerinti sorrend $\sigma_1 = (1, 4, 3, 5, 2)$, ehhez $p_B = (1, 5, 3, 2, 4)$ és $\sigma_2 = (1, 4, 5, 3, 2)$, ehhez $p_B = (1, 5, 4, 2, 3)$.

3.1.4. Egy randomizált algoritmus

A [2] cikkben szereplő KwikSort algoritmus egy randomizált algoritmus, amely élsúlyozatlan turnamenteken várható értékben 3-közelítő megoldást ad a FAS problémára. Az algoritmus során vesszük a turnament egy véletlen v csúcsát, és a többi csúcsot két csoportra osztjuk a következők szerint. A V_L csoportba kerülnek azon u csúcsok, amelyekből vezet egy uv él a v csúcsba, a V_R csoportba pedig a maradék w csúcsok, amelyekbe vezet vw él a v csúcsból. Ezek után az algoritmust rekurzívan meghívjuk a V_L csúcsok által feszített részgráfra és a V_R csúcsok által feszített részgráfra, majd visszatérünk azzal a csúcssorrenddel, amelyet a V_L csúcsait az első rekurzív hívás szerinti sorrendben a v elé, a V_R csúcsait a második rekurzív hívás szerinti sorrendben a v után írva kapunk.

Az algoritmus speciálisan élsúlyozott, oda-vissza irányított teljes gráfokon is konstans-közelítő megoldást ad a súlyozott FAS problémára. Egy $D = (V, E)$ élsúlyozott, oda-vissza irányított gráfból először készítünk egy élsúlyozatlan turnamentet a következő módon: Minden (u, v) csúcspárra behúzzuk uv és vu közül a nagyobb súlyú élt, egyenlőség esetén tetszőlegeset. Ezután az így kapott turnamenten futtatjuk az előző bekezdésben leírt KwikSort algoritmust. A [2] cikkben bizonyítják, hogy ez a módszer várható értékben 5-közelítő, ha a w élsúlyozás nemnegatív és minden u, v 2 V párra teljesül, hogy $w(uv) + w(vu) = 1$. Ha ezek mellett az élsúlyozásra a háromszög-egyenlőtlenség is teljesül, akkor várható értékben 2-közelítő. Ebből következik, hogy az algoritmus a $P_{\sum}$ feladatra is legalább 2-közelítő, mivel a büntetési gráf teljesíti a fenti feltételeket és láttuk, hogy a $P_{\sum}$ feladat ekvivalens a büntetési gráfon értelmezett súlyozott FAS feladattal. A cikkben adnak egy javított algoritmust is a $P_{\sum}$ feladatra, mely szerint ha vesszük a

KwikSort algoritmusból kapott permutáció és a PickAPerm$_{\sum}$ algoritmusból kapott permutáció közül azt, amelyre kisebb a $K_{\sum}$ távolság, akkor az így kapott permutáció várható értékben $\frac{11}{7}$-közelítő a $P_{\sum}$ problémára.

A KwikSort algoritmust a [32] cikkben derandomizálják. A derandomizált algoritmus során minden lépésben azon v csúcs szerint osztjuk két csoportra a csúcsokat, amelyre a $\frac{\sum_{uv2T_v\ (V)} \frac{8}{5}w(uv) \quad \frac{6}{5}\frac{w(uv)^2}{w(uv)+w(vu)}}{\sum_{uv2T_v\ (V)} w(vu)}$ érték minimális, ahol $T_v\ (V)$ jelöli a V csúcshalmazon a v csúcs szerinti kettéosztás miatt keletkező balra élek halmazát.

A cikkben belátják, hogy ha az így kapott permutáció és a PickAPerm$_{\sum}$ algoritmusból kapott permutáció közül vesszük azt, amelyikre a $K_{\sum}$ érték kisebb, akkor $\frac{8}{5}$-közelítő megoldást kapunk a $P_{\sum}$ feladatra.

3.1.5. Polinomiális approximációs séma

A [24] cikkben adnak egy randomizált és egy determinisztikus PTAS-t a FAS problémára élsúlyozott oda-vissza irányított gráfokon, ha a w élsúlyozás nemnegatív és minden (u, v) csúcspárra $w(uv) + w(vu) = 1$ teljesül. A PTAS három különböző típusú algoritmust használ fel: 1) egy konstans-közelítő algoritmust, amely lehet a korábban leírt KwikSort vagy Borda-módszer, 2) lokális javításokat és 3) egy additív approximációt biztosító algoritmust, amelyet a FAS probléma komplementerére, a maximum aciklikus részgráf feladatra vezetnek be a [3, 19] cikkekben.

A fejezet hátralévő részében áttekintjük a PTAS determinisztikus változatát. Adott ϵ mellett a gráf élsúlyait $\frac{}{n^2}$ egész többszörösére kerekítjük, majd a kapott gráfon kiszámolunk egy konstans-közelítő p permutációt. Ezután a következő két lépést iteráljuk, ameddig valamelyikkel tudunk javítani az aktuális p permutáción:

1. Lokális javítás: kivesszük a v versenyzőt a p által meghatározott sorrendből, és visszatesszük a i-edik helyre.
2. Additív approximáció: valamilyen $i \quad j$ indexekre tekintsük p permutáció szerint az i és j helyezések közötti csúcsokat. Az ezen csúcsok által feszített részgráfra alkalmazzuk a [3, 19] cikkekben leírt AddApprox algoritmus derandomizált változatát ϵ-tól függő additív hibával. A p permutációban az i és j helyezések közötti csúcsok sorrendjét módosítsuk az AddApprox algoritmusból kapott sorrendre.

Mivel az élsúlyokat $\frac{}{n^2}$ egész többszörösére kerekítettük, ezért minden javításnál legalább $\frac{}{n^2}$ értékkel javul a FAS súlyösszege. Az élsúlyozás tulajdonságai miatt a FAS költsége mindig legfeljebb n^2, tehát legfeljebb $\frac{n^4}{}$ javító lépést végezhetünk. Lokális javítások és az AddApprox hívások száma legfeljebb $\frac{n^6}{}$.

A cikkben bizonyítják, hogy ez az algoritmus $(1 + \epsilon)$-közelítő megoldást ad. Ehhez a PTAS egy randomizált változatát elemzik, amelyre itt nem térünk ki.

3.2. A $P_{\max}$ feladat

Ebben a fejezetben a $P_{\max}$ feladatra mutatunk egy 2-közelítő algoritmust a [8] cikk alapján, majd $k = 2$ bíró esetén egy szomszédos seréken alapuló optimális megoldást adó algoritmust a [29] cikk alapján, és egy hasonló közelítő algoritmust $k = 3$ bíró esetén.

3.2.1. A legjobb input permutáció

A PickAPerm$_{\max}$ algoritmus során veszünk egy tetszőleges $p_i \in P$ input permutációt a $P = \{p_1, \dots, p_k\}$ halmazból. A [8] cikk alapján megmutatjuk, hogy a PickAPerm$_{\max}$ algoritmus 2-közelítő a $P_{\max}$ feladatra.

3.9. Tétel. *A $P = \{p_1, \dots, p_k\}$ halmazon a $P_{\max}$ feladatra bármelyik $p_i \in P$ input permutáció 2-közelítő.*

Bizonyítás. A bizonyítást a p_1 permutációra mondjuk el, de ugyanígy működik tetszőleges input permutációra. Jelölje p^* az optimális közös permutációt és legyen p_{j^*} egy olyan input permutáció, amelynek legnagyobb a távolsága a p^* közös permutációtól. Tehát az optimális távolságra teljesül, hogy $OPT_{\max} = K(p_{j^*}, p^*)$.

Ekkor teljesül a következő becslés:

$$K_{\max}(P, p_1) = \max_{i=1,\dots,k} K(p_i, p_1) \leq \max_{i=1,\dots,k} K(p_i, p^*) + P(p^*, p_1) \leq 2K(p_{j^*}, p^*) = 2OPT_{\max},$$

ahol az első egyenlőtlenség abból következik, hogy a Kendall-τ távolságra teljesül a háromszög-egyenlőtlenség, a második egyenlőtlenség és a két egyenlőség pedig a p_{j^*} permutáció választása és az 1.5. Definíció miatt igaz.

Megfigyelés. A fenti bizonyítás csak a háromszög-egyenlőtlenséget használja ki, tehát ha a $P_{\max}$ feladatot a Kendall-τ távolság helyett másik távolságfogalommal definiáltuk volna, amelyre igaz a háromszög-egyenlőtlenség (például Spearman footrule távolsággal), akkor is működne a fenti bizonyítás. A $P_{\max}$ feladatot különböző távolságfogalmak szerint a [4] cikkben vizsgálják.

3.2.2. Lokális javítás $k = 2$ bíróra

Mutatunk egy polinomiális algoritmust a $P_{\max}$ feladatra $k = 2$ bíró esetén a [29] cikk alapján, amely szomszédos versenyzők megcserélésén alapul.

Algoritmus $k = 2$ bíróra:
Kiindulunk $p = p_1$ közös permutációból. Ha a p permutáció szerint léteznek olyan u és v szomszédos versenyzők, hogy u és v sorrendjének megcserélésével $K_{\max}(P, p)$ távolság csökken, akkor cseréljük meg u és v sorrendjét. Ezt ismételjük, amíg már nem létezik ilyen javító csere.

3.10. Állítás. *A fenti algoritmus $k = 2$ bíróra a $P_{\max}$ feladatra optimális megoldást ad.*

Bizonyítás. A kezdeti p választása miatt az összes keresztezés számára teljesül, hogy $K_{\max}(P,p) = K(p_1,p_2) = OPT_{\sum}(P)$. Az algoritmus során csak olyan u,v versenyzőkre létezhet javító csere, akik p_1 és p_2 permutációkban fordított sorrendben szerepelnek. Tehát minden javító cserével pontosan egy keresztezés kerül át a p_2 permutációtól a p_1 permutációhoz, ezért a $K_{\max}(P,p)$ távolság pontosan eggyel csökken, az összes keresztezés száma pedig állandó, azaz a $K_{\sum}(P,p) = OPT_{\sum}(P)$ egyenlőség végig teljesül.

Ha $p \neq p_2$, akkor az 1.2. Állítás szerint léteznek u,v szomszédos versenyzők a p permutáció szerint, hogy u és v között keresztezés van p_2 permutációban. Ebből következik, hogy az algoritmus csak akkor akadhat el, amikor az összes keresztezés lényegében egyenlően oszlik el p_1 és p_2 között, azaz

$$K(p_1,p) = \left\lfloor \frac{OPT_{\sum}(P)}{2} \right\rfloor \text{ és } K(p_2,p) = \left\lceil \frac{OPT_{\sum}(P)}{2} \right\rceil.$$

Ekkor $K_{\max}(P,p) = \left\lceil \frac{OPT_{\sum}(P)}{2} \right\rceil$, ebből következik, hogy p optimális az 1.6. Lemma miatt.

3.2.3. Lokális javítás $k = 3$ bíróra

Adunk egy szomszédos versenyzők cseréjén alapuló új közelítő algoritmust a $P_{\max}$ feladatra $k = 3$ bíró esetén.

Algoritmus:
A $P_{\sum}$ feladatra létezik PTAS [24]. Legyen a kezdeti p permutáció az innen kapott $(1+\epsilon)$-közelítő megoldás a $P_{\sum}$ feladatra . A $k = 2$ bíró esetéhez hasonlóan, ha léteznek p permutáció szerint szomszédos u,v versenyzők, hogy a megcserélésükkel $K_{\max}(P,p)$ csökken, akkor megcseréljük őket. Ezt ismételjük, amíg már nem létezik ilyen javító csere.

3.11. Állítás. *Az algoritmus futása után a kapott p megoldás költségére igaz, hogy*

$$K_{\max}(P,p) \quad \left\lceil \left(\frac{3}{2} + \epsilon\right) OPT_{\max}(P) \right\rceil.$$

Bizonyítás. Jelölje C az összes keresztezés számát kezdetben, tehát $C := K_{\sum}(P,p)$.

3.12. Állítás. *Az algoritmus futása után $K_{\max}(P,p) \quad \left\lceil \frac{C}{2} \right\rceil$.*

Bizonyítás. Feltehető, hogy a p_3 permutációnál van a legtöbb keresztezés. Az algoritmus csak akkor akadhat el, amikor p_1 vagy p_2 felé lényegében ugyanannyi keresztezés megy, mint p_3 felé. Feltehető, hogy p_2 miatt álltunk le. Kezdetben $K(p_2,p) + K(p_3,p) \quad C$, mert C jelölte az összes keresztezés számát. Az algoritmus futása közben $K(p_3,p)$ minden lépésben eggyel csökken, $K(p_2,p)$ pedig legfeljebb eggyel nő, ezért végig teljesül, hogy $K(p_2,p) + K(p_3,p) \quad C$. Tehát leálláskor teljesül, hogy

$$K(p_2,p) \quad \left\lfloor \frac{C}{2} \right\rfloor \text{ és } K(p_3,p) \quad \left\lceil \frac{C}{2} \right\rceil.$$

Ebből következik, hogy $K_{\max}(P,p) \quad \left\lceil \frac{C}{2} \right\rceil$.

Leálláskor a min-max távolságra igaz, hogy

$$K_{\max}(P,p) \quad \left\lceil \frac{C}{2} \right\rceil \quad \left\lceil \frac{(1+\epsilon)}{2} OPT_{\sum}(P) \right\rceil \quad \left\lceil \frac{3(1+\epsilon)}{2} OPT_{\max}(P) \right\rceil,$$

ahol az első egyenlőtlenség éppen az előző állítás, a második egyenlőtlenség kezdeti p választásából következik, a harmadik pedig az 1.6. Lemma állításából következik.

3.13. Következmény. *Ha $OPT_{\max}(P)$ tart a végtelenbe, akkor a közelítő hányados tart $\left(\frac{3}{2}+\epsilon\right)$-hoz. Ez azért hasznos, mert kis $OPT_{\max}(P)$ értékekre létezik FPT algoritmus.*

4. Egzakt módszerek

Ebben a fejezetben áttekintünk néhány egzakt megoldást adó, azonban általános esetben nem polinomiális futásidejű módszert. Ezek azért lényegesek, mert egy valódi verseny végeredményének meghatározásánál a résztvevők valószínűleg nem lennének elégedettek egy közelítő megoldással.

4.1. Egészértékű programozási modellek

Ismertetjük a $P_{\sum}$ feladat és a $P_{\max}$ feladat IP modelljét.

A $P_{\sum}$ feladat IP modellje több cikkben is előfordul [2, 11, 32]. A változók a versenyzőpároknak felelnek meg és az $x_{u<v}$ változó értéke 1, ha az u versenyző megelőzi a v versenyzőt a közös permutáció szerint, különben 0. Ezen változókkal a $P_{\sum}$ feladat IP modellje:

$$\min \sum_{u;v2V} A_P(v,u)x_{u<v} + A_P(u,v)x_{v<u} \qquad (IP_{\sum})$$

$$x_{u<v}\ 2\ f0,1g \qquad 8u,v\ 2\ V \qquad (8a)$$

$$x_{u<v} \quad x_{v<u} = 1 \qquad 8u,v\ 2\ V \qquad (8b)$$

$$x_{u<w} \quad x_{u<v} \quad x_{v<w} \quad 1 \qquad 8u,v,w\ 2\ V, \qquad (8c)$$

ahol a (8b) és a (8c) feltételek biztosítják, hogy a kapott $x_{u<v}$ változók tényleg egy permutációt határozzanak meg. Szokás szerint $A_P(u,v)$ jelöli azon permutációk számát az inputban, amelyek szerint az u versenyző megelőzi a v versenyzőt.

A [2] cikkben adnak a KwikSort algoritmus alapján egy randomizált, rekurzív algoritmust, amely során kerekítésekkel az LP relaxált egy megengedett megoldásából kapunk egy egészértékű megoldást. Ezen algoritmus bizonyítja, hogy az egészségi hézag legfeljebb $\frac{4}{3}$, azonban nyitott kérdés, hogy ez a becslés éles-e. A [32] cikkben derandomizálják az algoritmust.

A $P_{\sum}$ feladat IP modellje alapján könnyen felírható a $P_{\max}$ feladat IP modellje. Az $x_{u<v}$ változók mellé bevezetünk egy új $\lambda \geq 0$ változót, amely a közös permutáció maximális távolságának felel meg, ezt szeretnénk minimalizálni.

$$\min \lambda \qquad (IP_{\max})$$

$$x_{u<v} \in \{0,1\} \qquad \forall u,v \in V \qquad (9a)$$

$$x_{u<v} + x_{v<u} = 1 \qquad \forall u,v \in V \qquad (9b)$$

$$x_{u<w} \geq x_{u<v} + x_{v<w} - 1 \qquad \forall u,v,w \in V \qquad (9c)$$

$$\sum_{p_i(u)>p_i(v)} x_{u<v} \leq \lambda \qquad \forall p_i \in P. \qquad (9d)$$

A $P_{\sum}$ és a $P_{\max}$ feladatok IP modelljeit implementáltuk Python nyelven. A program forráskódja elérhető online [1].

4.2. Fix paraméteres algoritmusok

Áttekintjük a legfontosabb FPT algoritmusokat a $P_{\sum}$ feladatra a [7] cikk alapján, és a $P_{\max}$ feladatra a [29] cikk alapján.

Egyik problémára sem létezhet FPT algoritmus a bírók k számát paraméternek tekintve, ha P $\neq$ NP, mivel a 2. Fejezetben láttuk, hogy már $k = 4$ bíró esetén is NP-nehéz mindkét probléma. Helyette természetes paraméterek a versenyzők n száma, az optimális távolság c értéke, két input permutáció d_a átlagos távolsága és a maximális r_{max} távolság egy adott versenyző két helyezése között.

A versenyzők számát paraméternek tekintve

A versenyzők számával paraméterezve mind a $P_{\sum}$, mind a $P_{\max}$ feladatra könnyen adható FPT algoritmus, mivel n versenyző esetén összesen $n!$ különböző permutáció létezik és egy adott permutációnak a P halmaztól vett $K_{\sum}$, illetve $K_{\max}$ távolsága kiszámolható $O(kn \log n)$ lépésben. Tehát az összes lehetséges permutációt végigpróbálhatjuk $O(n!kn \log n)$ lépésben.

A $P_{\sum}$ feladatra létezik egy hatékonyabb algoritmus is [7], a továbbiakban ezt ismertetjük. Az algoritmus dinamikus programozáson alapul, amely a versenyzők minden $V' \subseteq V$ részhalmazára kiszámol egy optimális sorrendet, melyhez az eggyel kevesebb elemű részhalmazok optimális sorrendjeit használja fel. A V' részhalmaz optimális sorrendjéhez tekintsük az összes olyan V'' részhalmazt, melyre $V'' = V' \setminus \{v\}$, valamely $v \in V'$ versenyzőre. Adott V' esetén legfeljebb n különböző V'' részhalmaz létezhet. A V' részhalmazra az optimális sorrend meghatározásához tekintsük minden $v \in V'$ versenyző esetén azt a sorrendet, melyben az első helyen a v versenyző végzett, majd a $V'' = V' \setminus \{v\}$ részhalmaz versenyzői következnek az optimális sorrendben. Könnyen látható, hogy ezen sorrendek közül a minimális költségű sorrend optimális a V' részhalmazra. Egy adott részhalmazra

$O(kn^2)$ lépésben kiszámítható az optimum és összesen 2^n részhalmaza van a versenyzők halmazának, tehát az algoritmus lépésszáma $O(2^n kn^2)$.

Az optimális távolságot paraméternek tekintve

Az optimális távolsággal paraméterezett FPT algoritmust a $P_{\sum}$, illetve $P_{\max}$ feladatok eldöntési verzióira adjuk meg. A feladatok eldöntési verziói során nem optimális permutációt keresünk, hanem szeretnénk eldönteni, hogy létezik-e olyan közös permutáció, melynek az input permutációktól vett $K_{\sum}$, illetve $K_{\max}$ távolsága legfeljebb egy c szám.

Konstruálni fogunk a $P_{\sum}$ feladat eldöntési verziójára egy legfeljebb $2c$ versenyzőt és legfeljebb $2c$ permutációt tartalmazó kernelt. Amíg létezik olyan v versenyző az inputban, amelynek minden permutáció szerint ugyanaz a sorrendje minden más u versenyzővel szemben, addig töröljük ezeket a versenyzőket az inputból. Az 1.7. Lemmából következik, hogy ezen versenyzőknek minden optimális megoldásban egyértelmű a helyezése. Ha már nem tudunk több versenyzőt törölni, és több, mint $2c$ versenyző maradt, akkor „nem” a válasz, mivel tetszőleges p közös permutáció szerint minden maradék versenyző szerepel legalább egy keresztezésben valamelyik input permutáció felé. Ha létezik olyan p_i permutáció, amely c-nél többször szerepel az inputban, akkor minden p_i-től különböző p permutációra $K_{\sum}(P, p) > cK(p, p_i) > c$ teljesül. Tehát csak p_i lehet megengedett megoldás a feladatra, ezért ha $K(p_i, P) \quad c$, akkor „igen” a válasz, különben „nem”. Ezután minden permutációból legfeljebb c másolat van. Ha több mint $2c$ permutáció van, akkor „nem” a válasz, mivel tetszőleges közös permutációra létezik legalább $c + 1$ tőle különböző permutáció az inputban. Ezzel kaptunk egy legfeljebb $2c$ versenyzőt és legfeljebb $2c$ permutációt tartalmazó kernelt a feladatra. A kernelt megoldhatjuk az előző bekezdésben adott versenyzők számával paraméterezett FPT algoritmust felhasználva, vagy a [7] cikkben megadott hatékonyabb, keresőfa alapú algoritmust, amellyel a futásidő $O(1,53^c + kn^2)$.

$P_{\max}$ feladat eldöntési verziójára a [29] cikkben adnak két keresőfa alapú FPT algoritmust az optimális távolsággal paraméterezve, amelyekből az első algoritmust fogjuk részletesen leírni. A keresőfa minden csúcsához egy p permutáció és egy x egész szám tartozik, és olyan p megoldást keresünk, amelyre $K(p, p\ ) \quad x$. A gyökérhez egy tetszőleges p_i 2 P input permutáció tartozik és a c paraméter, amelyre meg szeretnénk oldani a $P_{\max}$ feladat eldöntési változatát. Ha egy (p, x) csúcsban $x < 0$, akkor az adott ágon nincs megoldás és nem építjük tovább a fát. Ha $K(p, p_i) \quad c$ minden p_i 2 P permutációra, akkor p egy megoldás ezért a válasz „igen”. Különben vegyünk egy p_i 2 P permutációt, amelyre $K(p, p_i) > c$. Ekkor létezik legalább $c + 1$ keresztezés p és p_i között, vegyünk ezek közül tetszőleges $c + 1$ versenyzőpárt és mindegyikhez tartozzon egy csúcs a keresőfa következő szintjén. A csúcsokhoz tartozzon az a permutáció, melyet a p_i permutációból az adott keresztezéshez tartozó két versenyző megcserélésével kapunk és csökkentsük eggyel x értékét. Az így gyártott keresőfában minden csúcsnak legfeljebb $c + 1$ gyereke van, és legfeljebb c szintje, mivel minden szinten eggyel csökkentettük a c értékét, és negatív c-re nem folytattuk az eljárást. Ha a feladatra „igen” a válasz, akkor a keresőfa valamely csúcsához rendelt p permutáció is egy megoldás, a továbbiakban ezt bizonyítjuk. Ha a (p, x) csúcsra létezik olyan p permutáció, amelyre $K(p, p\ )\quad x$ és $K_{max}(P, p\ )\quad c$, akkor a p permutációnál meghatározott c + 1 keresztezés közül legalább az egyik fordított sorrendben szerepel p szerint mint p szerint. Tehát létezik olyan keresztezés, amelyhez tartozó versenyzőket megcserélve a kapott p^0 permutációra $K(p^0, p\ ) < K(p, p\ )$, azaz a csúcsnak létezik olyan gyereke, amellyel közelebb kerülünk a megoldáshoz. Tehát ha létezik egy p megoldás, akkor a keresőfában létezik egy út, amellyel eljutunk eddig a megoldásig. A keresőfa mélysége legfeljebb c és minden csúcsának legfeljebb c + 1 gyereke van, ezért a futásidő $O(c^c kn \log n)$. A cikkben adnak egy hatékonyabb $O(2^{4c}kn^2 + kn^2 \log n)$ futásidejű algoritmust is.

Két input permutáció közötti átlagos távolságot paraméternek tekintve

Két input permutáció közötti átlagos távolságot jelölje d_a. Legyen $d := dd_a e$ és egy v versenyző átlagos helyezését jelölje $h_a(v)$. Megmutatjuk, hogy bármely a $P_{\sum}$ feladatra optimális p permutáció szerint minden v versenyző helyezésére teljesül a következő egyenlőtlenség: $h_a(v)\quad d < p\ (v) < h_a(v) + d$. Ez abból következik, hogy az optimális távolság értéke kevesebb mint kd, mivel létezik olyan input permutáció, amelyre kisebb a $K_{\sum}$ érték. Ha egy p permutációban létezik olyan v versenyző, amelynek legalább d távolságra van a helyezése az átlagos helyezésétől, akkor teljesül a következő becslés:

$$K_{\sum}(P, p) \quad \sum_{i=1}^{k} jp(v) \quad p_i(v)j \quad \Big|\sum_{i=1}^{k}\big(p(v) \quad p_i(v)\big)\Big| =$$

$$\Big|kp(v) \quad \sum_{i=1}^{k} p_i(v)\Big| = kjp(v) \quad h_a(v)j \quad kd,$$

ahol az első egyenlőtlenség abból következik, hogy a jobb oldala alsó becslés a v versenyző keresztezéseinek számára, amely alsó becslés a $K_{\sum}$ távolságra. A második egyenlőtlenség a háromszög-egyenlőtlenség, a két egyenlőség egyszerű algebrai átalakításokból következik, az utolsó egyenlőség pedig abból a feltevésből, hogy a v versenyző p szerinti helyezése legalább d távolságra van átlagos helyezésétől. Tehát ha létezik egy permutációban ilyen v versenyző, akkor a $K_{\sum}$ távolsága legalább kd, ezért nem lehet optimális.

Jelölje V_i azon versenyzők halmazát, akikre létezik olyan optimális permutáció, amely szerint az i-edik helyen végeztek. A korábbi $h_a(v)\quad d < p\ (v) < h_a(v) + d$ egyenlőtlenségből következik, hogy minden v versenyző csak $2d\quad 1$ szomszédos helyezésre kerülhet, tehát $jV_ij\quad 4d$ teljesül minden i helyezésre. Innen kiszámolható egy optimális permutáció dinamikus programozással, amelynek a lényegét ismertetjük vázlatosan. Minden i helyezésre kipróbáljuk az összes lehetséges versenyzőt a V_i halmazból. Ha egy adott v versenyző kerül az i-edik helyre, akkor a maradék versenyzőket a $V_i\, n\, fvg$ részhalmazból kettéoszthatjuk az alapján, hogy v elé vagy v mögé kerülnek a sorrendben. Egy három dimenziós T táblázat $T(i, v, V_i^0)$ mezőjében kiszámoljuk az első i helyezés versenyzőit tartalmazó keresztezések minimális számát, ha a v versenyző végzett az i helyen és a V_i' halmaz versenyzői végeztek előtte a V_i halmazból. Ezt minden lehetséges $v \in V_i$ versenyzőre és $V_i' \subseteq V_i \setminus \{v\}$ részhalmazra kiszámoljuk, az $(i-1)$ helyezésnél kiszámolt értékeket felhasználva. A futásidő $O(16^d(d^2n + dkn^2 \log n))$.

A $P_{\max}$-ra nem létezhet FPT algoritmus d_a-val paraméterezve, ha P $\neq$ NP, mivel az átlagos távolságot tetszőlegesen lecsökkenthetjük egy input permutáció sokszori hozzávételével, azonban az így kapott inputra a megoldás megegyezik az eredeti input optimális megoldásával.

Adott versenyző két helyezése közötti maximális távolságot paraméternek tekintve

A maximális távolságot jelölje $r_{\max}$, azaz

$$r_{\max} = \max_{p_i,p_j \in P} \max_{v \in V} |p_i(v) - p_j(v)| + 1.$$

Ha egy u versenyző megelőzi a v versenyzőt minden input permutáció szerint, akkor az 1.7. Lemma szerint minden $P_{\sum}$ feladat szerinti optimális sorrendben is. Tekintsünk egy tetszőleges u versenyzőt, akinek a minimális helyezése a, a maximális helyezése b. Ekkor egy v versenyzőnek akkor kérdéses az u versenyzővel szemben a helyezése, ha létezik olyan $p_i \in P$ permutáció, mely szerint $a < p_i(v) < b$. Ekkor a v versenyző helyezésére minden input permutációban igaz, hogy $a - r_{\max} < p_i(v) < b + r_{max}$, különben nagyobb lenne az r_{max} távolság. Tehát minden versenyzőnek legfeljebb $3r_{\max}$ másik versenyzővel lehet kérdéses a sorrendje, azaz minden versenyző legfeljebb $3r_{\max}$ szomszédos hely valamelyikére kerülhet egy optimális sorrendben. Ebből következik, hogy minden helyezésre azon versenyzők száma akik optimális megoldásban kerülhetnek az adott pozícióra legfeljebb $6r_{\max}$. Innen a d_a paraméteres esethez hasonlóan dinamikus programozással kiszámolható egy optimális permutáció.

A $P_{\max}$ feladatra a [29] cikkben belátják, hogy már $r_{\max} = 1$ esetén sem létezik polinomiális algoritmus, ha P $\neq$ NP.

5. A $P^v_{\max}$ feladat

Ebben a fejezetben felvetünk egy új problémát, a $P^v_{\max}$ feladatot. A $P_{\max}$ feladatban a bírók szerint szerettük volna minimalizálni a maximális távolságot, most a versenyzők szemszögéből nézzük a feladatot:

Azt mondjuk, hogy egy v versenyző elégedetlen, ha a bírók többségénél v jobb helyezésen végzett, mint u, de a közös sorrendben v rosszabb helyezésen végzett, mint u. Továbbra is $A_P(u,v)$ jelöli minden (u,v) versenyzőpárra, hogy a P halmaz permutációi közül u hányban végzett jobb helyezésen, mint v. Ezzel a jelöléssel egy v versenyző elégedetlensége egy p közös permutáció szerint

$$e_p(v) = |\{u : A_P(v,u) > A_P(u,v) \text{ és } p(u) < p(v)\}|,$$

azaz azon versenyzők száma, akik hamarabb végeztek a sorrendben mint v, pedig a bírók többségénél v végzett jobb helyen. Adott P halmazra definiálhatunk egy alternatív büntetési gráfot, melynek csúcshalmaza megegyezik a versenyzők halmazával és az u, v csúcsok között akkor vezet egy uv irányított él, ha $A_P(u, v) > A_P(v, u)$. Ezt a gráfot megkaphatjuk az 1.3. Fejezetben definiált büntetési gráf átalakításával, ha az oda-vissza élek helyett csak a nagyobb súlyú élt húzzuk be élsúlyozás nélkül (egyenlőség esetén egyik irányban sem vezet él).

5.1. Definíció. *A p permutációnak a P halmaztól vett versenyzők szerinti min-max távolsága a maximálisan elégedetlen versenyző elégedetlensége, azaz*

$$K^v_{\max}(P, p) = \max_{v \in V} e_p(v).$$

A $P^v_{\max}$ feladatnál olyan p közös permutációt keresünk, amelyre a $K^v_{\max}(P, p)$ távolság minimális.

Szemléletesen az alternatív büntetési gráfban szeretnénk egy olyan csúcssorrendet találni, amely minimalizálja a maximális bal kifok értékét, ahol egy csúcs bal kifokának az adott csúcsból a sorrend szerint őt megelőző csúcsokba kiinduló élek számát nevezzük. Egy v csúcs egy adott sorrend szerinti bal kifokát $\overleftarrow{\delta}(v)$ jelöli, egy D' részgráfra megszorított kifokát pedig $\delta_{D'}(v)$ jelöli.

A $P^v_{\max}$ feladatot vizsgálhatjuk általánosan irányított gráfokon, azaz szeretnénk egy adott D irányított gráfra egy olyan csúcssorrendet találni, amely minimalizálja a maximális bal kifok értékét. Erre a feladatra megadunk egy polinom idejű algoritmust.

1. Algoritmus MinMaxBalKifok($D = (V, E)$)

1: $D'(V', E') := D(V, E)$
2: $n := |V|$
3: Jelölje $\sigma_1, \dots, \sigma_n$ a keresett csúcssorrendet
4: **for** $i = n, \dots, 1$ **do**
5: $\quad \sigma_i := \arg\min_{v \in V'} \{\delta_{D'}(v)\}$
6: $\quad D' := D' \setminus \{\sigma_i\}$
7: **end for**
8: **output** $(\sigma_1, \dots, \sigma_n)$

Az 1. Algoritmus során jobbról balra rögzítjük a csúcsok sorrendjét, D' jelöli a még nem rögzített csúcsok által feszített részgráfot. Az algoritmus 1. sorában vesszük a D' részgráf egy minimális kifokú csúcsát, rögzítjük a sorrend utolsó még szabad helyére és töröljük a gráfból. Ezt ismételjük, amíg minden csúcsot rögzítettünk.

Az algoritmus helyességének a bizonyításához szükségünk lesz a következő állításra.

5.2. Állítás. *Egy D irányított gráfban tetszőleges csúcssorrend szerinti maximális bal kifok értékre alsó korlát egy tetszőleges D' feszített részgráfban a minimális kifok.*

Bizonyítás. Legyen σ egy tetszőleges csúcssorrend a D irányított gráf csúcsain. Tekintsük a D' részgráf csúcsainak σ szerinti sorrendjében az utolsó csúcsot, jelölje ezt a csúcsot v. Ennek a v csúcsnak minden D' részgráfon belüli ki-éle balra megy, tehát az összes csúcs tetszőleges sorrendjében a bal kifoka legalább akkora lesz, mint a D' részgráfra megszorított kifoka, azaz $\overleftarrow{\delta}(v) \geq \delta_{D'}(v)$.

5.3. Állítás. *Az 1. Algoritmus megad egy olyan csúcssorrendet, amely minimalizálja a maximális bal kifok értékét.*

Bizonyítás. Jelölje $\sigma_1, \dots, \sigma_n$ az algoritmus által megtalált csúcssorrendet, és legyen σ_i az a csúcs, amelynek a bal kifoka maximális ezen sorrend szerint. Legyen D' a $\sigma_1, \dots, \sigma_i$ csúcsok által feszített részgráf — amely egybeesik az algoritmus szerinti D' gráffal σ_i rögzítésekor. Ebben a D' gráfban σ_i egy minimális kifokú csúcs, mert amikor rögzítettük a σ_i csúcs helyét, akkor pontosan a D' részgráf csúcsai közül választhattunk az algoritmus 5. sorában. Tehát a megoldásban a maximális bal kifok értéke megegyezik a D' részgráfból kapott alsó korláttal, így a megoldás optimális az 5.2. Állítás miatt.

6. Egy sorbarendezési feladat

Ebben a fejezetben a $P^v_{\max}$ feladat eldöntési verziójának általánosításait vizsgáljuk.

A V halmazból a v elem elhagyásával kapott halmazt jelölje $V - v$.

6.1. Definíció (**($f, g; h$)-tulajdonságú sorrend**)**.** *Egy V alaphalmaz minden v elemére adott egy $h_v : 2^{V-v} \to \mathbb{R}$ monoton növő halmazfüggvény, azaz tetszőleges $A \subseteq B \subseteq V - v$ esetén $h_v(A) \leq h_v(B)$ teljesül. Adott egy $f : V \to \mathbb{R}$ alsó korlát és egy $g : V \to \mathbb{R}$ felső korlát. Azt mondjuk, hogy a V alaphalmaz egy σ sorrendje $(f, g; h)$-tulajdonságú, ha minden v elemre teljesül, hogy*

$$f(v) \leq h_v(\overleftarrow{\sigma}(v)) \leq g(v),$$

ahol $\overleftarrow{\sigma}(v)$ jelöli a σ sorrendben a v elemet megelőző elemek halmazát. Tehát σ olyan sorrend, hogy minden v elemre, az őt megelőző elemek halmazának h_v szerinti értéke a v elemhez tartozó alsó és felső korlát közé esik.

Azon speciális esetben, amikor az elemekhez nincs alsó korlát, csak a $g : V \to \mathbb{R}$ felső korlát adott, a felső korlátot teljesítő σ sorrendeket $(-\infty, g; h)$-tulajdonságúnak nevezzük. Hasonlóan, amikor nincs felső korlát, csak az $f : V \to \mathbb{R}$ alsó korlát adott, az alsó korlátot teljesítő σ sorrendeket $(f, \infty; h)$-tulajdonságúnak nevezzük.

Az $(f, g; h)$-tulajdonságú sorrend feladat az előző fejezetben definiált $P^v_{\max}$ feladat eldöntési verziójának általánosítása. A h_v függvény rendelje hozzá egy adott $V' \subseteq V - v$ halmazhoz, azon $u \in V'$ versenyzők számát, akiket a bírók többségénél megelőz a v versenyző, azaz $h_v(V') := |\{u \in V' : A_P(v, u) > A_P(u, v)\}|$. Ekkor $g \equiv c$ esetben $(-\infty, g; h)$-tulajdonságú sorrend feladatnál azt kell eldöntenünk, hogy létezik-e olyan sorrend, melyben minden versenyző elégedetlensége legfeljebb c. Ez a feladat éppen a $P^v_{\max}$ feladat eldöntési verziója.

6.1. Csak felső korlát: $(\quad 1 , g; h)$-tulajdonságú sorrend

A következő állításban mutatunk egy sértő halmazt $(\quad 1 , g; h)$-tulajdonságú sorrend létezésére, melyre szükségünk lesz a 6.3. Tétel bizonyításához.

6.2. Állítás. *Ha a V alaphalmaznak létezik olyan V^0 részhalmaza, hogy V^0 minden v elemére igaz a $h_v(V^0 \quad v) > g(v)$ egyenlőtlenség, akkor nem létezik $(\quad 1 , g; h)$-tulajdonságú sorrend a V alaphalmazon.*

Bizonyítás. Tekintsük a V alaphalmaz elemeinek egy tetszőleges σ sorrendjét. Vegyük a σ sorrend szerinti utolsó elemét a V^0 részhalmaznak, jelölje ezt az elemet v. Ekkor a v elemre teljesül az alábbi egyenlőtlenség:

$$h_v(\overleftarrow{\sigma}(v)) \quad h_v(V^0 \quad v) > g(v),$$

ahol az első egyenlőtlenség azért igaz, mert v a σ sorrend szerinti utolsó eleme a V^0 részhalmaznak, ezért a $V^0 \quad v$ halmaz részhalmaza a $\overleftarrow{\sigma}(v)$ halmaznak és h_v monoton növő halmazfüggvény. A második egyenlőtlenség a V^0 részhalmaz definíciójából következik. A fenti egyenlőtlenségből látszik, hogy a σ sorrendben a v elemet megelőző elemek halmazához rendelt h_v érték sérti a g felső korlátot, így a σ sorrend nem lehet $(\quad 1 , g; h)$-tulajdonságú.

Adunk egy algoritmust $(\quad 1 , g; h)$-tulajdonságú sorrend létezésének eldöntésére.

2. Algoritmus $(\quad 1 , g; h)$-TULAJDONSÁG

```
 1: V0 := V
 2: n := jVj
 3: Jelölje σ1, ..., σn a keresett sorrendet
 4: for i = n, ..., 1 do
 5:     V  := fv 2 V0 : hv(V0   v)   g(v)g
 6:     if V  6= ; then
 7:         Legyen σi 2 V  tetszőleges
 8:         V0 := V0   σi
 9:     else
10:         output Nincs megoldás
11:         exit
12:     end if
13: end for
14: output σ1, ..., σn
```

A 2. Algoritmus jobbról balra rögzíti az elemek sorrendjét, közben V^0 jelöli a még nem rögzített elemek részhalmazát . Az algoritmus 7. sorában mindig vesszük a V^0 részhalmaz egy olyan v elemét, amelyre $h_v(V^0 \quad v) \quad g(v)$ teljesül, azaz a többi még nem rögzített elem részhalmazához rendelt h_v érték legfeljebb akkora, mint a v elemhez tartozó felső korlát. Ezt az elemet rögzítjük a σ sorrendben az utolsó még szabad helyre és töröljük a

V^0 részhalmazból. Ha valamikor nem létezik ilyen v elem, akkor kiírjuk, hogy nem létezik $(\quad 1\,, g; h)$-tulajdonságú sorrend és az algoritmus leáll.

6.3. Tétel. *A 2. Algoritmus polinom sok lépésben, a h függvényt legfeljebb $O(n^2)$ részhalmazon kiértékelve eldönti, hogy a V alaphalmazon létezik-e $(\quad 1\,, g; h)$-tulajdonságú sorrend.*

Bizonyítás. Az algoritmus 7. sorában mindig olyan elemet rögzítünk a sorrendben, amely teljesíti a hozzá tartozó felső korlátot. Ha minden lépésben létezett ilyen elem, akkor az algoritmus által meghatározott $\sigma_1, \ldots, \sigma_n$ sorrend $(\quad 1\,, g; h)$-tulajdonságú. Az algoritmus csak akkor akadhat el, ha az 5. sorában definiált $V\quad$ halmaz üres, azaz a még nem rögzített elemek V^0 halmazában minden v elemre teljesül a $h_v(V^0 \quad v) > g(v)$ egyenlőtlenség. Tehát V^0 egy sértő halmaz a 6.2. Állítás miatt, így tényleg nem létezik a keresett sorrend.

A 2. Algoritmus helyességéből következik a 6.2. Állítás erősebb változata:

6.4. Tétel. *A V alaphalmazon pontosan akkor létezik $(\quad 1\,, g; h)$-tulajdonságú sorrend, ha nem létezik olyan V^0 részhalmaz, melynek minden v elemére igaz a $h_v(V^0 \quad v) > g(v)$ egyenlőtlenség.*

6.2. Csak alsó korlát: $(f,\ 1\,; h)$-tulajdonságú sorrend

A következő állításban mutatunk egy sértő halmazt $(f,\ 1\,; h)$-tulajdonságú sorrend létezésére, melyre szükségünk lesz a 6.6. Tétel bizonyításához.

6.5. Állítás. *Ha a V alaphalmaznak létezik olyan V^0 részhalmaza, hogy V^0 minden v elemére $h_v(V n V^0) < f(v)$ egyenlőtlenség teljesül, akkor nem létezik $(f,\ 1\,; h)$-tulajdonságú sorrend a V alaphalmazon.*

Bizonyítás. Tekintsük a V alaphalmaz elemeinek egy tetszőleges σ sorrendjét. Vegyük a V^0 részhalmaznak a σ sorrend szerinti első elemét, jelölje ezt az elemet v. Ekkor a v elemre teljesül a következő egyenlőtlenség:

$$h_v(\overleftarrow{\sigma}(v)) \quad h_v(V\, n\, V^0) < f(v),$$

ahol az első egyenlőtlenség azért igaz, mert v a σ sorrend szerinti első eleme a V^0 részhalmaznak, ezért $\overleftarrow{\sigma}(v)$ részhalmaza a $V n V^0$ halmaznak és h_v monoton növő halmazfüggvény. A második egyenlőtlenség a V^0 részgráf definíciójából következik. Tehát a σ sorrend szerint a v elemet megelőző elemek részhalmazához tartozó h_v érték sérti az f alsó korlátot, így a σ sorrend nem lehet $(f,\ 1\,; h)$-tulajdonságú.

Adunk egy algoritmust $(f,\ 1\,; h)$-tulajdonságú sorrend létezésének eldöntésére.

3. Algoritmus $(f, \infty; h)$-TULAJDONSÁG

1: $V' := V$
2: $n := |V|$
3: Jelölje $\sigma_1, \ldots, \sigma_n$ a keresett sorrendet
4: **for** $i = 1, \ldots, n$ **do**
5: $\quad V^* := \{v \in V' : h_v(V \setminus V') \geq f(v)\}$
6: $\quad$ **if** $V^* \neq \emptyset$ **then**
7: $\qquad$ Legyen $\sigma_i \in V^*$ tetszőleges
8: $\qquad V' := V' - \sigma_i$
9: $\quad$ **else**
10: $\qquad$ **output** Nincs megoldás
11: $\qquad$ **exit**
12: $\quad$ **end if**
13: **end for**
14: **output** $\sigma_1, \ldots, \sigma_n$

A 3. Algoritmus balról jobbra rögzíti az elemek sorrendjét, szemben a 2. Algoritmussal. Közben V' jelöli a még nem rögzített elemek részhalmazát. Az algoritmus 7. sorában mindig vesszük a V' halmaz egy olyan v elemét, amelyre $h_v(V \setminus V') \geq f(v)$ teljesül, azaz a már rögzített elemek részhalmazához tartozó h_v érték legalább akkora, mint az $f(v)$ alsó korlát. Ezt a v elemet rögzítjük a sorrend következő helyére és töröljük a V' részhalmazból. Ha valamikor nem létezik ilyen v elem, akkor kiírjuk, hogy nem létezik $(f, \infty; h)$-tulajdonságú sorrend és az algoritmus leáll.

6.6. Tétel. *A 3. Algoritmus polinom sok lépésben, a h függvényt legfeljebb $O(n^2)$ részhalmazon kiértékelve eldönti, hogy a V alaphalmazon létezik-e $(f, \infty; h)$-tulajdonságú sorrend.*

Bizonyítás. Az algoritmus 7. sorában mindig egy olyan elemet rögzítünk a sorrendben, amely teljesíti a hozzá tartozó alsó korlátot. Ha minden lépésben létezett ilyen elem, akkor a $\sigma_1, \ldots, \sigma_n$ sorrend $(f, \infty; h)$-tulajdonságú. Az algoritmus csak akkor akadhatott el, ha az 5. sorában definiált V^* halmaz üres, azaz a még nem rögzített elemek V' részhalmazában minden v elemre teljesül a $h_v(V \setminus V') < f(v)$ egyenlőtlenség, tehát V' egy sértő halmaz a 6.5. Állítás miatt, így tényleg nem létezik a keresett sorrend.

A 3. Algoritmus helyességéből következik a 6.5. Állítás erősebb változata:

6.7. Tétel. *A V alaphalmazon pontosan akkor létezik $(f, \infty; h)$-tulajdonságú sorrend, ha nem létezik olyan V' részhalmaz, melynek minden v elemére teljesül a $h_v(V \setminus V') < f(v)$ egyenlőtlenség.*

6.3. Bal súlyozott kifok korlátos csúcssorrend

Ebben a fejezetben az $(f, g; h)$-tulajdonságú sorrend fogalmát értelmezzük irányított gráfokon, speciális h halmazfüggvényekre.

Adott egy $D = (V, E)$ irányított hurokélmentes gráf, a $w : E \to \mathrm{R}_+$ élsúlyozással. Adott egy $f : V \to \mathrm{R}_+$ alsó korlát és egy $g : V \to \mathrm{R}_+$ felső korlát. Definiáljuk a $h_v : 2^{V - v} \to \mathrm{R}_+$ halmazfüggvényt az alábbi módon:

$$h_v(V') = \sum_{e \in \delta(v, V')} w(e),$$

ahol $\delta(v, V')$ jelöli a v csúcsból a V' csúcshalmazba menő élek halmazát. Az így kapott h_v függvény az élsúlyozás nemnegativitása miatt monoton növő. Ekkor $h_v(\overleftarrow{\sigma}(v)) = \overleftarrow{\delta}^w(v)$, ahol $\overleftarrow{\delta}^w(v)$ jelöli a sorrend szerinti bal súlyozott kifokát a v csúcsnak. Ezen h_v szerint az $(f, g; h)$-tulajdonságú sorrendek éppen azon σ sorrendek, melyekben minden v csúcsra teljesül, hogy

$$f(v) \le \overleftarrow{\delta}^w(v) \le g(v),$$

azaz a csúcsok σ sorrend szerinti bal súlyozott kifoka legalább akkora, mint a csúcshoz tartozó alsó korlát és legfeljebb akkora, mint a csúcshoz tartozó felső korlát.

A feladat egy ekvivalens átfogalmazása, hogy a D élsúlyozott irányított gráfban szeretnénk eldönteni létezik-e egy olyan Feedback Arc Set, amelyben minden csúcs bal súlyozott kifoka teljesíti az f és g korlátokat. Ezért a feladatot $(f, g; \sum w)$-FAS problémának nevezzük a Feedback Arc Set feladat után, élsúlyozatlan gráfra pedig (f, g)-FAS problémának.

Kimondjuk a 6.4. Tételt és a 6.7. Tételt az $(f, g; \sum w)$-FAS probléma csak alsó, illetve csak felső korlátos esetének megoldhatóságára.

6.8. Tétel. *A D élsúlyozott irányított gráfban pontosan akkor létezik $(-\infty, g; \sum w)$-FAS problémára megengedett megoldás, ha nem létezik olyan $D' = (V', E')$ feszített részgráf, hogy D' minden v csúcsára $\delta^w(v, V') > g(v)$ teljesül, ahol $\delta^w(v, V')$ jelöli a v csúcsnak a D' részgráfra megszorított súlyozott kifokát.*

6.9. Tétel. *A D élsúlyozott irányított gráfban pontosan akkor létezik $(f, \infty; \sum w)$-FAS problémára megengedett megoldás, ha nem létezik olyan $D' = (V', E')$ feszített részgráf, hogy D' minden v csúcsára $\delta^w(v, V \setminus V') < f(v)$ teljesül, ahol $\delta^w(v, V \setminus V')$ jelöli a v csúcsnak a $V \setminus V'$ csúcshalmazba kimenő éleire megszorított súlyozott kifokát.*

Megfigyelés. Az $(f, \infty; \sum w)$-FAS probléma ekvivalens a $(-\infty, g; \sum w)$-FAS problémával, mert egy σ sorrend pontosan akkor teljesíti az g felső korlátot, ha a σ sorrend fordítottja teljesíti a $(\delta^w - g)$ alsó korlátot. Azonban általános esetben az $(f, g; h)$-tulajdonságú sorrendeknél nem működik ez a visszavezetés, ezért volt szükséges az alsó- és a felső korlátos esetet külön tárgyalni a 6.1 és 6.2 Fejezetekben.

Vegyük észre, hogy a csak felső korlátos $(-\infty, g, \sum w)$-FAS feladat akkor is megoldható, ha adott a csúcshalmazon egy R részbenrendezés, és olyan felső korlátot teljesítő csúcssorrendet keresünk, amely nem sérti a részbenrendezést. Ez a probléma visszavezethető a $(-\infty, g; \sum w)$-FAS feladatra, ha minden (u, v) csúcspárra, melyeknél a részbenrendezés szerint az u csúcs megelőzi a v csúcsot hozzáveszünk a gráfhoz egy uv élt, és beállítjuk a $w(vu)$ súlyt az u csúcshoz rendelt felső korlátnál nagyobbra. Ekkor megengedett sorrendekben az uv él csak jobbra él lehet, tehát az u csúcs megelőzi a v csúcsot, így a megengedett sorrendek nem sértik a részbenrendezést.

6.3.1. Alkalmazások

Mutatunk egy alkalmazást a csak felső korlátos feladatra: A D gráfon a $(-\infty, g)$-FAS feladatra $g \equiv c$ konstans felső korláttal pontosan akkor létezik megengedett σ csúcssorrend, ha a D irányított gráf felbontható egy minden csúcsban legfeljebb c kifokszámú és egy aciklikus gráf uniójára. Tetszőleges megengedett sorrend szerinti jobbra élek az aciklikus gráf élei, a balra élek pedig a kifok korlátos részgráf élei. Vegyük észre, hogy fordítva ez nem teljesül, azaz nem létezik minden felbontáshoz egy megengedett sorrend, melyben a felbontás egyik komponense a balra élek részgráfja, másik komponense a jobbra élek részgráfja. Viszont az igaz, hogy minden felbontásban az aciklikus gráf tetszőleges topologikus sorrendje egy megengedett sorrend (lehet, hogy a sorrend szerint kerülnek plusz élek az aciklikus gráfba). A továbbiakban a $c \equiv 1$ esettel foglalkozunk.

6.10. Tétel. *Egy $D = (V, E)$ irányított gráfon polinom időben eldönthető, hogy adott $X \subseteq V$ csúcshalmazra létezik-e olyan $B \subseteq E$ befenyves, amelynek minden $v \in X$ csúcs gyökere, és a D irányított gráf előáll a B befenyves és egy aciklikus gráf uniójaként. Pontosan akkor létezik ilyen felbontás, ha nem létezik olyan $D' = (V', E')$ feszített részgráf a D gráfban, amelyben minden $v \in X$ csúcs kifoka legalább 1 és minden $v \in V' \setminus X$ csúcs kifoka legalább 2.*

Bizonyítás (6.10. Tétel). Konstruálunk a feladathoz egy $(-\infty, g)$-FAS feladatot a D gráfon. A $v \in X$ csúcsokon legyen $g(v) = 0$, a $v \in V \setminus X$ csúcsokon pedig $g(v) = 1$ a felső korlát. Megmutatjuk, hogy erre a $(-\infty, g)$-FAS feladatra pontosan akkor létezik megoldás, ha létezik olyan B befenyves, amelynek minden $v \in X$ csúcs gyökere és a D gráf előáll a B befenyves és egy aciklikus gráf uniójaként. Ha létezik egy σ megengedett sorrend, akkor legyen az aciklikus gráf a σ sorrend szerinti jobbra éleket tartalmazó gráf, B pedig a σ sorrend szerinti balra éleket tartalmazó gráf. A felső korlátok miatt B tényleg egy befenyves, amelynek minden $v \in X$ csúcs gyökere. Fordított irányban, ha létezik megengedett felbontás, akkor legyen σ a felbontás szerinti aciklikus gráfnak egy tetszőleges topologikus sorrendje. Ekkor σ megengedett sorrend a $(-\infty, g)$-FAS feladatra, mert a balra élek halmaza a B befenyves éleinek részhalmaza, így minden csúcs teljesíti a hozzá tartozó felső korlátot. A 6.8. Tételt alkalmazva kapjuk pontosan akkor létezik ilyen felbontás, ha nem létezik olyan D' feszített részgráf, amelyben minden $v \in X$ csúcs kifoka legalább 1 és minden $v \in V \setminus X$ csúcs kifoka legalább 2.

6.11. Következmény. *Speciálisan $X = \emptyset$ esetben egy $D = (V, E)$ irányított gráf pontosan akkor áll elő egy $B \subseteq E$ befenyves és egy aciklikus gráf uniójaként, ha nem létezik olyan $D' = (V', E')$ feszített részgráf a D gráfban, amelyben minden $v \in V'$ csúcs kifoka legalább 2.*

Megjegyezzük, hogy a 6.10. Tételben a B befenyvesnek lehetnek $v \notin X$ gyökerei is, ezért a tételből nem következik, hogy adott D irányított gráfról eldönthető, hogy előáll-e egy befenyő és egy aciklikus gráf uniójaként.

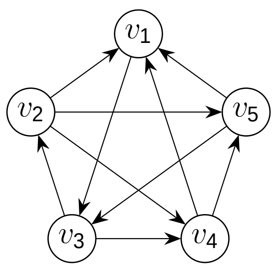
A 6.10. Tételből következik az alábbi tétel.

6.12. Következmény. *Egy $D = (V, E)$ irányított gráfon polinom időben eldönthető, hogy adott $B \subseteq E$ befenyvesnek létezik-e olyan $B' \subseteq E$ szintén befenyves kiegészítése, hogy D előáll a B' befenyves és egy aciklikus gráf uniójaként. Pontosan akkor létezik ilyen kiegészítés, ha nem létezik olyan $D' = (V', E')$ feszített részgráf a D gráfban, amelyben minden $v \in V'$ csúcs kifoka legalább $2 - \delta_B(v, V \setminus V')$.*

Bizonyítás. Töröljük a D gráfból a B befenyves éleit, jelölje az így kapott gráfot D'. Legyen X azon v csúcsok halmaza, melyekből kiindult él a B befenyvesben. Ekkor az eredeti D gráfban a B befenyvesnek pontosan akkor létezik a tétel szerinti B' kiegészítése, ha a D' gráf felbomlik egy aciklikus gráfra és egy olyan befenyvesre, amelyben minden $v \in X$ csúcs gyökér. Erre a feladatra a 6.10. Tételt alkalmazva épp a bizonyítandó állítást kapjuk.

6.4. Linking tulajdonság

Az (f, g)-FAS feladatra nem teljesül a linking tulajdonság, azaz abból, hogy létezik az f alsó korlátra megengedett sorrend és létezik a g felső korlátra megengedett sorrend nem következik, hogy létezik olyan sorrend, amely egyszerre megengedett az f alsó és a g felső korlátra is (feltéve hogy $f \leq g$).



3. Ábra. Ellenpélda, amely bizonyítja, hogy az (f, g)-FAS problémára nem teljesül a linking tulajdonság.

Tekintsük a 3. Ábrán adott irányított súlyozatlan teljes gráfot. Legyen $g \equiv 1$ a felső korlát és $f(v_1) = 0$, a többi v_i csúcsra $f(v_i) = 1$ az alsó korlát. Ebben a példában a g felső korlátra megengedett megoldás a v_2, v_3, v_4, v_5, v_1 sorrend, az f alsó korlátra megengedett megoldás a v_1, v_2, v_3, v_4, v_5 sorrend, de nem létezik f alsó és g felső korlátra egyszerre megengedett megoldás. Azért nem létezhet ilyen, mert csak a v_1 csúcs kifoka nem nagyobb mint a felső korlátja, tehát minden g szerint megengedett sorrendben a v_1 csúcs az utolsó, viszont minden f szerinti megengedett sorrendben csak a v_1 csúcs lehet az első, mert minden más csúcsnak legalább 1 az alsó korlátja.

Mivel az (f, g)-FAS feladat az $(f, g; h)$-tulajdonságú sorrend feladat speciális esete, ezért az $(f, g; h)$-tulajdonságú sorrend feladatra sem teljesül a linking tulajdonság. Ezzel szemben a 6.1 és a 6.2 Fejezetekben tárgyalt csak alsó- és csak felső korlátos eseteket felhasználva könnyen bizonyítható, hogy a linking tulajdonság teljesül a következő esetben:

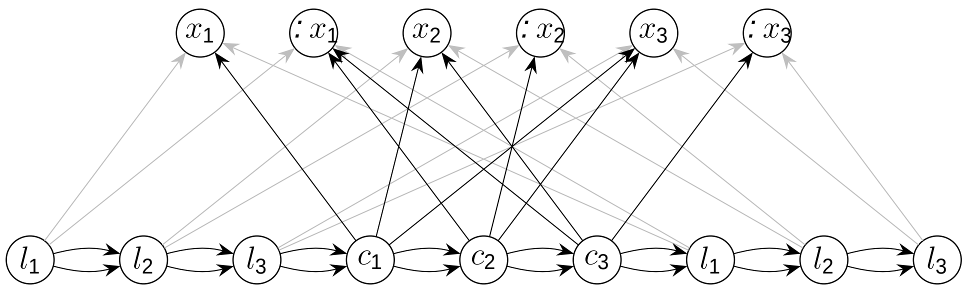
6.13. Állítás. *Ha minden csúcson vagy csak alsó, vagy csak felső korlát adott, akkor teljesül a linking tulajdonság.*

A következő fejezetben látni fogjuk, hogy ez a feltétel nem gyengíthető, azaz ha létezik egyetlen elem, amelyre egyszerre adott alsó és felső korlát is, akkor a probléma NP-teljes.

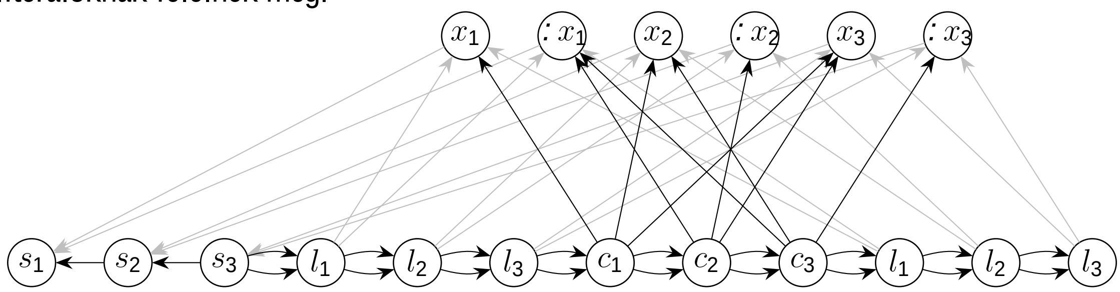
6.5. Nehézségi eredmények

A következő tételben kimondjuk, hogy az (f,g)-FAS probléma NP-teljes, már minden csúcsra konstans kifokszámú gráfokon is.

6.14. Tétel. *Az (f,g)-FAS probléma NP-teljes az $f \leq g$ esetben. A feladat már akkor is nehéz, ha a gráf minden csúcsának a foka legfeljebb 7, és az $f \leq g$ korlát egy csúcsban 0, a többiben 1.*



4. Ábra. A 6.14. Tétel bizonyításában a visszavezetés során készített gráf szemléltetése a következő konjunktív normálformára $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$. A gráfban a c_i csúcsok a klózoknak, l_i csúcsok a változóknak, az x_i és $\neg x_i$ csúcsok a literáloknak felelnek meg.



5. Ábra. A 6.14. Tétel bizonyításában a $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$ konjunktív normálformához készített gráf kiegészítve minden változóhoz az s_i csúccsal.

Bizonyítás. Világos, hogy az (f,g)-FAS feladat NP-beli. A 3-XSAT-3 feladatban adott egy konjunktív normálforma, amelyben minden klóz pontosan 3 literált tartalmaz és minden változó pontosan 3 klózban szerepel. Szeretnénk eldönteni, hogy kielégíthető-e az adott konjunktív normálforma úgy, hogy minden klózban pontosan egy literál igaz. Ismert, hogy a 3-XSAT-3 feladat NP-teljes [27]. Ezt a problémát fogjuk visszavezetni a majdnem minden csúcsra azonosan 1 korlátos (f, g)-FAS feladatra, amihez a következők szerint konstruálunk meg egy (f, g)-FAS feladatot. A D irányított gráfban a konjunktív normálforma minden literáljának megfelel egy x_i vagy $:x_i$ csúcs (a 4. Ábra felső sora). Minden klóznak megfelel egy c_i csúcs, valamint minden változónak megfelel két l_i csúcs. Minden c_i klózból megy egy él a benne szereplő literálok x_i vagy $:x_i$ csúcsába és minden l_i változónak megfelelő csúcsból kiindul két él az adott változóhoz tartozó x_i és $:x_i$ literáloknak megfelelő csúcsokba. Tekintsük a 4. Ábra alsó sorában az $l_1, \ldots, l_n, c_1, \ldots, c_k, l_1, \ldots, l_n$ csúcsokat. A sorrend szerint minden csúcsból megy két párhuzamos él a következőbe. Belátható, hogy a 4. Ábrán szemléltetett D irányított gráfban az alsó sor csúcsain azonosan 1, a felső sor csúcsain azonosan 0 korlátokat megkövetelve pontosan akkor megoldható az (f, g)-FAS feladat, ha a 3-XSAT-3 feladat megoldható az adott konjunktív normálformára. Ezt az állítást nem bizonyítjuk, de később mutatunk egy hasonló bizonyítást az 5. Ábrán szemléltetett konstrukcióra. Ahhoz, hogy a felső csúcsokra is azonosan 1 korlátot követelhessünk meg, hozzáveszünk az alsó csúcsok elé minden változóhoz egy s_i csúcsot. Az így kibővített gráfot az 5. Ábrán szemléltetjük. Minden x_i és $:x_i$ literál csúcsából kiindul egy él az adott literálhoz tartozó s_i változó csúcsába. Minden s_i csúcsból megy egy él az őt megelőző s_i csúcsba, valamint az utolsóból hozzáveszünk két párhuzamos élt az első l_1 csúcsba, ezzel biztosítjuk, hogy az 5. Ábra alsó sorában lévő csúcsok meghatározott sorrendben szerepeljenek minden megengedett sorrendben. Legyenek f g 1 korlátok, kivéve az s_1 csúcsra, amelyre $f(s_1) = g(s_1) = 0$.

A konjunktív normálformának pontosan akkor létezik 3-XSAT-3 feladatra megoldása, ha a D gráfra definiált (f, g)-FAS feladat megoldható. Vegyük a konjunktív normálforma egy megoldását. Az alsó sor csúcsai legyenek az ábra szerinti $l_1, \ldots, l_n, c_1, \ldots, c_k, l_1, \ldots, l_n$ sorrendben, az igaz literálokat tegyük közvetlenül az s_i változók csúcsai mögé, a hamis literálokat tegyük a sorrend végére, így megengedett sorrendet kapunk a D gráfon definiált (f, g)-FAS feladatra. Fordított irányban, ha veszünk egy megengedett (f, g)-FAS megoldást, akkor a c_i klózoknak megfelelő csúcsok előtti literálokat igaznak választva, a többi literált hamisnak választva a konjunktív normálforma egy megoldását kapjuk, a c_i klózokhoz tartozó $f(c_i) = g(c_i) = 1$ korlátok miatt. A literálokat beállíthatjuk így, mert minden i indexre az x_i és a $:x_i$ literálok közül pontosan az egyik szerepel a korábbi l_i változó előtt és a másik a későbbi l_i változó mögött, tehát nem lesz olyan változó, amelyre x_i és $:x_i$ literált egyszerre igaznak vagy egyszerre hamisnak állítjuk. A D gráfban minden x_i vagy $:x_i$ literálhoz tartozó csúcs és minden l_i változóhoz tartozó csúcs foka legfeljebb 6, minden c_i csúcs foka legfeljebb 7 és minden s_i csúcs foka legfeljebb 4. Tehát a D irányított gráfban minden csúcs foka legfeljebb 7.

Az (f, g)-FAS probléma az $(f, g; h)$-tulajdonságú sorrend feladat egy speciális esete, ezért a 6.14. Tételből következik az alábbi tétel.

6.15. Tétel. *Az $(f, g; h)$-tulajdonságú sorrend létezésének eldöntése NP-teljes az f g esetben, már akkor is, ha az f g korlát egy elemre 0, a többi elemre 1.*

A következő tételekben megmutatjuk, hogy az $(f, g; \sum w)$-FAS probléma és az (f, g)-FAS probléma egy-egy másik speciális esete is NP-teljes.

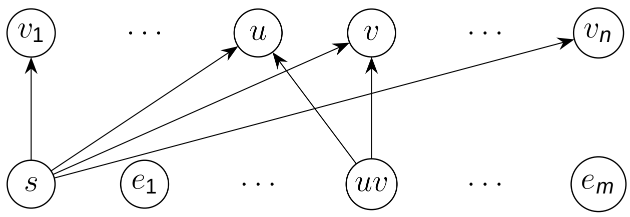
6.16. Tétel. *A* $(\quad 1\ , g; \sum w)$*-FAS probléma NP-teljes, ha megengedünk negatív* $w(e)$ *élsúlyokat.*

6.17. Következmény. *A* $(\quad 1\ , g; h)$*-tulajdonságú sorrend feladat is NP-teljes, ha nem követeljük meg a* h_v *halmazfüggvény monotonitását.*

6.18. Tétel. *Az* (f, g)*-FAS probléma NP-teljes, ha egy csúcsra alsó és felső korlát, a többi csúcsra csak felső korlát adott.*

6.19. Következmény. *Az* $(f, g; h)$*-tulajdonságú sorrend feladat is NP-teljes, ha létezik olyan elem, amelyre egyszerre adott alsó és felső korlát is.*

Tekintsük a 6. Ábrán szemléltetett konstrukciót, amelyet mind a 6.16. mind a 6.18. Tétel bizonyításánál fel fogunk használni.



6. Ábra. A 6.16. és 6.18. Tételek bizonyításakor konstruált gráf, amellyel visszavezetjük a feladatokra a k méretű független csúcshalmaz feladatot.

A k méretű független csúcshalmaz feladatról ismert, hogy NP-teljes [22], ezt vezetjük vissza mindkét feladatra. Legyen $G = (V, E)$ az a gráf, melyben el szeretnénk dönteni, hogy létezik-e k méretű független csúcshalmaz. Definiáljuk hozzá a következő D irányított gráfot: A G gráf minden v csúcsának és minden e élének megfelel a D gráf egy csúcsa, valamint D tartalmaz egy plusz s csúcsot is. A D irányított gráfban minden $e = uv$ élnek megfelelő e 2 V_D csúcsból kiindul két irányított él a végpontjainak megfelelő u 2 V_D és v 2 V_D csúcsokba. Emellett az s csúcsból vezet minden olyan v 2 V_D csúcsba egy él, amely az eredeti G gráf csúcsainak felel meg. A 6. Ábra alsó sorában lévő csúcsok sorrendjét rögzítjük, ezt a két feladatnál hasonlóan érjük el, új élek bevezetésével. Ezen konstrukciót felhasználva bizonyítjuk a 6.16. és 6.18. Tételeket.

Bizonyítás (6.16. Tétel). Látható, hogy a $(\quad 1\ , g; \sum w)$-FAS feladat NP-beli. Kiindulunk a konstrukció szerinti D gráfból, amelyet a 6. Ábra szemléltet. Az s csúcsból kiinduló élekhez $w(e) = \quad 1$, a többi élhez $w(e) = 1$ élsúlyokat rendelünk. Az s csúcshoz $g(s) = \quad k$, a többi csúcshoz $g(v) = 1$ felső korlátot rendelünk. Most bevezetjük az éleket, amelyek biztosítják, hogy a 6. Ábra alsó sorában lévő csúcsok minden megengedett sorrendben egymáshoz képest a megadott sorrendben szerepeljenek. Az alsó sorban az s csúcsból kiindul egy él az e_1 csúcsba, $w(se_1) = n$ súllyal, a többi csúcsból kiindul egy $w(e) = 2$ súlyú él, az őt követő csúcsba.

Ha a G gráfban létezett k méretű független csúcshalmaz, akkor létezik megengedett sorrend a D gráfon definiált $(\quad 1\ , g; \sum w)$-FAS feladatra. A σ sorrend elején legyenek a független csúcshalmaz csúcsai tetszőleges sorrendben, utánuk következnek az alsó sor csúcsai a meghatározott sorrendben, majd a többi csúcs. Az így kapott σ sorrend megengedett megoldás a D irányított gráfon definiált (1 $, g; \sum w$)-FAS feladatra, mert a felső sor csúcsainak 0 a kifoka, tehát minden sorrendben teljesítik a felső korlátot. Az s csúcs előtt pontosan k darab csúcs szerepel (ezért $\overleftarrow{\delta}(s) =$ k), az éleknek megfelelő csúcsokat pedig legfeljebb egyik végpontjuk előzi meg a σ sorrend szerint (ezért $\overleftarrow{\delta}(e) = 1$). Tehát az alsó sor csúcsai sem sértik a hozzájuk rendelt korlátokat.

Fordított irányban, ha veszünk egy σ megengedett sorrendet a (1 $, g; \sum w$)-FAS feladatra, akkor a σ sorrendben az alsó sor csúcsai biztosan a megadott sorrendben szerepelnek a köztük bevezetett felső korlátjuknál nagyobb súlyú élek miatt, tehát s megelőzi az összes élnek megfelelő e 2 V_D csúcsot. A $g(s) =$ k felső korlát miatt az s csúcsot legalább k darab v 2 V_G csúcs megelőzi a sorrendben. Mivel minden élnek megfelelő e 2 V_D csúcsra $g(e) = 1$ a felső korlát, ezért az él két végpontja közül legfeljebb az egyik lehet előtte. Ezekből következik, hogy a σ sorrendben az s csúcsot megelőző csúcsok halmaza egy legalább k méretű független csúcshalmaz a G gráfban, tehát a sorrend első k csúcsa éppen egy k méretű független csúcshalmaz.

Bizonyítás (6.18. Tétel). Könnyen látható, hogy az (f, g)-FAS feladat NP-beli. Kiindulunk a konstrukció szerinti D gráfból, amit a 6. Ábra szemléltet. Az s csúcsra $f(s) = g(s) = k$ korlátot rendelünk, a többi csúcsra $g(v) = 1$ felső korlátot rendelünk. Most bevezetjük az éleket, amelyek biztosítják a 6. Ábra alsó sorában lévő csúcsok rögzített sorrendjét. Az alsó sorban az s csúcsból az e_1 csúcsba kiindul n darab párhuzamos él, a többi csúcsból kiindul két párhuzamos él az őt követő csúcsba.

Ha a G gráfban létezik k méretű független csúcshalmaz, akkor legyen σ az a sorrend, ahol ezen csúcsok vannak elöl, aztán az alsó sor csúcsai a meghatározott sorrendben, majd a maradék csúcsok. Ekkor σ egy megengedett sorrend a D gráfon definiált (f, g)-FAS feladatra, mert a felső sor csúcsainak 0 a kifoka , tehát minden sorrendben teljesítik a hozzájuk tartozó korlátot. Az s csúcs előtt pontosan k darab csúcs van (ezért $\overleftarrow{\delta}(s) = k$), és az éleknek megfelelő csúcsokat legfeljebb egyik végpontjuk előzi meg a σ sorrendben (ezért $\overleftarrow{\delta}(e)$ 1). Tehát az alsó sor csúcsai is teljesítik hozzájuk tartozó korlátot.

Fordított irányban, ha létezik egy σ megengedett sorrend az (f, g)-FAS feladatra, akkor ebben a sorrendben az alsó sor csúcsai a megadott sorrendben szerepelnek, a köztük bevezetett párhuzamos élek miatt, tehát az s csúcs megelőzi az összes élnek megfelelő e 2 V_D csúcsot. Az $f(s) = g(s) = k$ korlát miatt pontosan k darab v 2 V_G csúcs van a sorrend szerint s előtt. Az éleknek megfelelő e 2 V_D csúcsokat a $g(e) = 1$ felső korlát miatt legfeljebb egyik végpontjuk előzheti meg. Ezekből következik, hogy a σ sorrendben az s csúcsot megelőző csúcsok halmaza egy k méretű független csúcshalmaz a G gráfban.

7. Nyitott kérdések

Zárásként mutatunk néhány nyitott kérdést a rangsorolási feladatok témakörében.

Először áttekintjük a $P_{\sum}$ és $P_{\max}$ feladatokkal kapcsolatos nyitott kérdéseket. Mindkét feladat nehézsége nyitott kérdés $k = 3$ bíró esetén, illetve a $P_{\sum}$ feladat nehézsége $k = 5$ bíró esetén is. Fontos megjegyezni, hogy a 2.2. Fejezetben mutatott visszavezetés miatt $k = 3$ bíró esetén, ha a $P_{\max}$ feladat megoldható, abból következik a $P_{\sum}$ feladat megoldhatósága, valamint ha a $P_{\sum}$ feladat NP-nehéz, abból következik a $P_{\max}$ feladat nehézsége.

A közelítő algoritmusokról szóló fejezetben láttuk, hogy a $P_{\sum}$ feladatra a Borda-módszer 5-közelítő, a KwikSort algoritmus és a PickAPerm$_{\sum}$ algoritmus közül a jobb megoldás választása pedig $\frac{11}{7}$-közelítő, azonban nem ismert pontos példa egyik algoritmusra sem. Nyitott kérdés, hogy a közelítő hányadosra adott becslések pontosak-e. Hasonló kérdés merül fel az egészségi hézag esetén is, amelyre a [2] cikkben megmutatják, hogy legfeljebb $\frac{4}{3}$, azonban pontos példa nélkül, tehát a becslés nem feltétlen éles.

A $P_{\sum}$ feladatra létezik polinomiális approximációs séma [24], de tudomásunk szerint nincs ismert polinomiális approximációs séma a $P_{\max}$ feladatra. Sőt általános k bíró esetén nem ismert 2-közelítőnél jobb algoritmus sem.

Az FPT algoritmusoknál láttuk, hogy a $P_{\max}$ feladatra nem létezik a d_a átlagos input távolsággal paraméterezett FPT algoritmus, ha $\mathrm{P} \neq \mathrm{NP}$, azonban ennek bizonyítása arra épül, hogy az inputhoz egy p_i permutációt tetszőlegesen sokszor hozzáveszünk. Kérdéses tehát, hogy ha nincs ismétlődés az inputban, akkor létezik-e FPT algoritmus a d_a átlagos input távolsággal paraméterezve.

A továbbiakban áttérünk az általunk bevezetett $(f, g; h)$-tulajdonságú sorrendhez kapcsolódó kérdésekre. Láttuk, hogy az (f, g)-FAS feladat NP-nehéz, már f g szigorú korlátok esetén is. Azonban nyitott kérdés, hogy tág korlátok esetén, például egy G élsúlyozatlan gráfon minden v csúcsra az $f(v) = 1$ alsó és a $g(v) = \delta(v)$ 1 felső korlátok esetén megoldható-e a feladat.

Az (f, g)-FAS feladat alkalmazásánál láttuk, hogy egy D irányított gráfról polinom időben eldönthető, hogy felbomlik-e egy befenyves és egy aciklikus gráf uniójára. Azonban nyitott kérdés, hogy befenyves helyett egy befenyőre, vagy még általánosabban egy adott gyökérhalmazú befenyvesre is megoldható-e a felbontási feladat.

Gondolkozhatunk az $(f, g; h)$ tulajdonságú sorrend létezése, illetve az (f, g)-FAS feladat általánosításain is. Például definiálhatjuk a költséges változatukat, amelynél minden elemhelyezés párra adott egy c költség, és egy legolcsóbb $(f, g; h)$-tulajdonságú sorrendet keresünk. Természetes módosítás az adott elemet megelőző összes elem helyett az f, g korlátokat az őt megelőző (legfeljebb) d elemre előírni. Egy másik lehetséges általánosítás, ha közös sorrend helyett egy kétdimenziós elhelyezésben gondolkodunk, az adott elemet „megelőző” elemek halmaza pedig a tőle balra-felfele elhelyezkedő elemeket tartalmazza.

Hivatkozások

[1] A Kendall-τ és a min-max Kendall-τ rangsorolás IP modelljének implementációja, `https://github.com/nborsik/konszenzusos-rangsorok`.

[2] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM (JACM)*, 55(5):1–27, 2008.

[3] Sanjeev Arora, Alan Frieze, and Haim Kaplan. A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 21–30. IEEE, 1996.

[4] Christian Bachmaier, Franz J. Brandenburg, Andreas Gleißner, and Andreas Hofmeier. On the hardness of maximum rank aggregation problems. *Journal of Discrete Algorithms*, 31:2–13, 2015.

[5] Georg Bachmeier, Felix Brandt, Christian Geist, Paul Harrenstein, Keyvan Kardel, Dominik Peters, and Hans G. Seedig. k-majority digraphs and the hardness of voting with a constant number of voters. *Journal of Computer and System Sciences*, 105:130–157, 2019.

[6] John Bartholdi, Craig A. Tovey, and Michael A. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and welfare*, 6(2):157–165, 1989.

[7] Nadja Betzler, Michael R. Fellows, Jiong Guo, Rolf Niedermeier, and Frances A. Rosamond. Fixed-parameter algorithms for Kemeny rankings. *Theoretical Computer Science*, 410(45):4554–4570, 2009.

[8] Therese Biedl, Franz J. Brandenburg, and Xiaotie Deng. On the complexity of crossings in permutations. *Discrete Mathematics*, 309(7):1813–1823, 2009.

[9] Duncan Black et al. *The theory of committees and elections*. Springer, 1958.

[10] Jean-Charles de Borda. Mémoire sur les élections au scrutin. *Histoire de l'Academie Royale des Sciences pour 1781 (Paris, 1784)*, 1784.

[11] Bryan Brancotte, Bo Yang, Guillaume Blin, Sarah Cohen-Boulakia, Alain Denise, and Sylvie Hamel. Rank aggregation with ties: Experiments and analysis. *Proceedings of the VLDB Endowment (PVLDB)*, 8(11):1202–1213, 2015.

[12] Arthur H. Copeland. A reasonable social welfare function. Technical report, mimeo, 1951. University of Michigan, 1951.

[13] Don Coppersmith, Lisa K. Fleischer, and Atri Rurda. Ordering by weighted number of wins gives a good ranking for weighted tournaments. *ACM Transactions on Algorithms (TALG)*, 6(3):1–13, 2010.

[14] Nicolas de Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix*. 1785.

[15] Persi Diaconis and Ronald L. Graham. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(2):262–268, 1977.

[16] Irit Dinur and Shmuel Safra. The importance of being biased. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 33–42, 2002.

[17] Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th international conference on World Wide Web*, pages 613–622, 2001.

[18] Guy Even, Baruch Schieber, Madhu Sudan, et al. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998.

[19] Alan Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999.

[20] Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the random ordering is hard: Inapproximability of maximum acyclic subgraph. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 573–582. IEEE, 2008.

[21] Viggo Kann. *On the approximability of NP-complete optimization problems*. PhD thesis, Citeseer, 1992.

[22] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[23] John G. Kemeny. Mathematics without numbers. *Daedalus*, 88(4):577–591, 1959.

[24] Claire Kenyon-Mathieu and Warren Schudy. How to rank with few errors. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 95–103, 2007.

[25] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 767–775, 2002.

[26] Christian Klamler. Borda and Condorcet: Some distance results. *Theory and Decision*, 59(2):97–109, 2005.

[27] Stefan Porschen, Tatjana Schmidt, Ewald Speckenmeyer, and Andreas Wotzlaw. XSAT and NAE-SAT of linear CNF classes. *Discrete Applied Mathematics*, 167:1–14, 2014.

[28] Donald G. Saari and Vincent R. Merlin. The Copeland method. *Economic Theory*, 8(1):51–76, 1996.

[29] Niko Schwarz. *Rank aggregation by criteria. Minimizing the maximum Kendall-tau distance*. PhD thesis, Diplomarbeit, Jena, 2009.

[30] Paul D. Seymour. Packing directed circuits fractionally. *Combinatorica*, 15(2):281–288, 1995.

[31] Nicolaus T. Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3):185–206, 1987.

[32] Anke van Zuylen and David P. Williamson. Deterministic algorithms for rank aggregation and other ranking and clustering problems. In *International Workshop on Approximation and Online Algorithms*, pages 260–273. Springer, 2007.