Eötvös Loránd Tudományegyetem

Természettudományi Kar

Vass Balázs

Online algoritmusok árverési feladatokban

BSc Szakdolgozat

Témavezetők:

Kovács Erika Renáta

ELTE, Operációkutatási Tanszék

Long Tran-Thanh

University of Southampton

Budapest, 2014

Köszönetnyilvánítás

Ezúton szeretném megköszönni Kovács Erika Renátának, hogy a témaválasztástól kezdve az apró részletekkel kapcsolatos észrevételekig kitartóan vezetett szakdolgozatom megalkotása alatt.

Köszönet Long Tran-Thanhnak az általa mutatott madártávlatért és az ismeretlent feszegető kérdéseiért és válaszaiért.

Köszönet mindazoknak, akik a matematika útját mutatták nekem. Köszönöm az egyetemnek és környezetemnek a légkört, amivel körbevett. Köszönöm a napnak, hogy sütött, az esőnek, hogy esett, a húroknak, hogy zengettek, a színeknek, hogy meglettek.

Örömmel végeztem a munkát.

Tartalomjegyzék

Előszó

Bizonyára többen találkoztak már Szindbád esetével a háremhölgyekkel, mely gyakran felbukkanó feladat a magyar szakirodalomban. Leírása a következő.

„Szindbád megmentette a kalifa életét, és ezért jutalmul feleségül veheti egyik háremhölgyét. A háremhölgyek sorban vonulnak el Szindbád előtt, egyszerre csak egyikőjük van jelen. Szindbád minden háremhölgy szépségét össze tudja hasonlítani a már elhaladottakéval, és egyértelműen meg tudja állapítani, hogy az eddig látott háremhölgyek közül ki a legszebb. Egy éppen megjelent háremhölgyről megjelenése után azonnal el kell döntse, hogy őt akarja-e feleségül venni, és ezt a döntést későbbb nem változtathatja meg. Szindbád tudja, hogy a kalifának hány háremhölgye van, viszont nem tudja, hogy a még nem látott háremhölgyek milyen szépek. A háremhölgyek véletlen sorrendben jelennek meg, és minden sorrend egyformán valószínű. Szindbád szeretné a legcsinosabb háremhölgyet választani. Milyen algoritmussal tudja ezt a lehetõ legnagyobb valószínűséggel elérni, és mekkora ez a valószínűség?"

A feladat megoldása szerint Szindbád számára az a legígéretesebb, ha elengedi a háremhölgyek bő egyharmadát, megjegyzi, milyen szép volt közülük a legszebb, majd a következőkben az első annál is csinosabbat kéri magának. Ekkor bő egyharmad eséllyel szerzi meg a legszebbet.

A Szindbád-probléma ismertségén kívül két szempontból is jó példa. Egyrészt *online* feladat abban az értelemben, hogy Szindbád még az adatok (azaz a háremhölgyek egymáshoz viszonyított szépsége) érkezése alatt, ám az összes adat ismerete nélkül kénytelen visszavonhatatlan döntéseket hozni, melyekről kiderülhet, hogy nem voltak tökéletesek. Mint érezzük, nem várhatjuk, hogy az online feladatokat „jobban" meg tudjuk oldani, mint az *offline* párjaikat (amikor az összes adat birtokában hozhatjuk döntéseinket). Példánk offline változatának megoldása nyilvánvaló: Szindbád összehasonlítja az őt körülülő háremhölgyek külcsíneit, majd kiválasztja magának a legszebbet.

Másrészt - egy kis fantáziával - Szindbád példája rávezet az árverési feladatokra. Képzeljük el, hogy Szindbád el szeretne venni egy olyan csinos háremhölgyet, akinek ő maga is rokonszenves. Tegyük fel továbbá, hogy Szindbád társaságában található Ali Baba és a

negyven rabló, akiknek Szindbáddal megegyező szándékuk van. Miközben a háremhölgyek egyesével vonulnak el előttük (így a kölcsönös szimpátiák időközben derülnek ki), céljuk minél több házasság köttetése. Természetesen amelyik hölgy elhaladt, már nem kérhető vissza.

E szakdolgozatban vizsgált árverési feladatok szintén online-ok. A következőképp lehet elképzelni őket. Egy árverésen jelen van néhány ügynök, akik a szerre érkező tárgyakra várnak. Minden ügynök esetén az általa éppen birtokolt tárgyak halmazának függvényében ismerjük, hogy mekkora hasznot hoz neki az éppen árverezendő tárgy, amennyiben hozzá kerül. Feladatunk az éppen érkező tárgy hozzárendelése egy ügynökhöz úgy, hogy az ügynökök összhasznát (a közjót) az árverés során maximalizáljuk. A kezelhetőség érdekében feltesszük, hogy egy új tárgy egy ügynöknek egy általa birtokolt bővebb tárgyahalmaz esetén legfennebb akkora hasznot hoz, mint egy szűkebb halmaz esetén. Az ügynökök haszonfüggvényeinek ezt a tulajdonságát fogjuk *szubmodularitásnak* nevezni.

Nyilvánvalóan léteznek nem szubmoduláris árverések is (például jegygyűrűk párban szoktak hasznot hozni), ezekről nem lesz szó a szakdolgozatban. Megjegyzendő továbbá, hogy nem foglalkozunk az árverések játékelméleti vonatkozásaival.

A szakdolgozat felépítése a következő. Az első fejezet rövid áttekintést ad a szubmoduláris függvények néhány tulajdonságáról, alkalmazási lehetőségéről és offline maximalizálásáról. A második fejezetben először bevezetjük a determinisztikus online algoritmusokat és a c-közelítőségük fogalmát, majd a lapozási feladat példájában alkalmazzuk ezeket. A második fejezet második felében bemutatjuk a véletlen online algoritmusokat, különböző ellenfeleiket, az algoritmusok különböző esetekben való c-közelítőségi fogalmait, majd a maximális párosítás online keresésén keresztül közelebbi képet alkotunk róluk. A harmadik fejezetben rátérünk az online szubmoduláris árverések és különböző sajátos eseteik vizsgálatára.

Ismeretes, hogy amennyiben az ügynökök haszonfüggvényei monoton szubmodulárisok, a mohó algoritmus által szolgáltatott megoldás biztosan eléri az optimum felét, azaz 1/2-közelítő. Tavalyi, 2013-as eredmény, hogy ebben az esetben nem létezik a mohónál lényegesen jobb, azaz $(1/2 + \epsilon)$-közelítő algoritmus (ahol $\epsilon > 0$). Long Tran-Thanh felvetése az volt, hogy vizsgáljuk meg a feladat speciális eseteit. Egy lehetséges eset az, amikor minden érkező tárgy egyforma az ügynökök haszonfüggvényei számára. Könnyű belátni, hogy ekkor a mohó algoritmus a feladat egy optimális megoldását szolgáltatja. Egy másik sajátos eset az, amikor az ügynökök haszonfüggvényei megegyeznek. Beláttuk, hogy a mohó algoritmus legfennebb 3/4-közelítő lehet. Nyitott kérdés azonban, hogy a közelítőségi hányadosa ténylegesen eléri-e ezt a számot, illetve, hogy létezik-e a mohónál hatékonyabb algoritmus.

1. fejezet

Szubmoduláris függvények

1.1. Szubmoduláris függvények és tulajdonságaik

A szubmodularitás halmazfüggvények tulajdonsága. Legyen a V alaphalmaz véges, legyen $f : 2^V \to \mathbb{R}$ halmazfüggvény, és $f(\emptyset) = 0$. Az egyik legkézenfekvőbb példa a szubmoduláris függvények szemléltetésére a következő. Tekintsük egy város vízvezeték-hálózatát, melybe időnként szennyeződés kerül. Szeretnénk érzékelőket telepíteni ezek mielőbbi észlelése érdekében. Legyen V azon helyek halmaza, ahova szerelhetők ezek, $f(S)$ pedig jelentse az S-beli pontokra helyezett érzékelők együttes hatékonyságát. A továbbiakban a szubmodularitás definiálása után megmutatjuk, hogyan is kerül elő a példánkban ez a tulajdonság.

1.1.1. Definíció. (Diszkrét derivált) *Egy $f : 2^V \to \mathbb{R}$ halmazfüggvény, $S \subseteq V$, és $e \in V$ esetén legyen a $\Delta_f(e|S) := f(S \cup \{e\}) - f(S)$ kifejezés az f-nek S-ben vett e irányú diszkrét deriváltja.*

Hasonlóan értelmezhető a halmaz irányú diszkrét derivált. Legyen f és S mint az előbb. Ekkor egy $T \subseteq V$ esetén legyen a $\Delta_f(T|S) := f(S \cup T) - f(S)$ kifejezés az f-nek S-ben vett T irányú diszkrét deriváltja.

Amennyiben az f függvény egyértelmű, gyakran elhagyjuk az alsó indexből, és egyszerűen a következőt írjuk: $\Delta(e|S)$.

1.1.2. Megjegyzés. $T \subseteq V$ esetén legyen t_k T k-adik eleme. Ekkor minden $S \subseteq V$-re

$$\Delta(T|S) = \sum_{k=1}^{|T|} \Delta(t_k|S \cup \{t_l | l \in 1, ..., k-1\}).$$

1.1.3. Definíció. (Szubmodularitás) *Egy $f : 2^V \to \mathbb{R}$ függvény szubmoduláris, ha*

$$\textit{minden } A \subseteq B \subseteq V, e \in V \setminus B - \textit{re } \Delta(e|A) \geq \Delta(e|B). \tag{1.1}$$

1.1.4. Állítás. *Egy $f : 2^V \to \mathbb{R}$ függvény pontosan akkor szubmoduláris, ha*

$$\text{minden } A, B \subseteq V - \text{re } f(A \cap B) + f(A \cup B) \leq f(A) + f(B). \tag{1.2}$$

Bizonyítás. Először azt mutatjuk meg, hogy (1.2) $\Rightarrow$ (1.1). Adott $A \subseteq B \subseteq V$ és $e \in V \backslash B$ esetén legyen $X = B$ és $Y = A \cup \{e\}$. Ekkor (1.2) miatt igaz, hogy

$$f(X \cap Y) + f(X \cup Y) \leq f(X) + f(Y)$$

$$f(A) + f(B \cup \{e\}) \leq f(B) + f(A \cup \{e\})$$

$$f(B \cup \{e\}) - f(B) \leq f(A \cup \{e\}) - f(A)$$

$$\Delta(e|B) \leq \Delta(e|A).$$

Most mutassuk meg, hogy (1.1) $\Rightarrow$ (1.2). Adott A és $B \subseteq V$ esetén legyen $X = A \cap B$ és $Y = A$. Mivel $X \subseteq Y$, 1.1.3 és (1.1) alapján $B \setminus A$ elemeit egyenként, X-hez és Y-hoz egyszerre véve azt kapjuk, hogy

$$\Delta(B \setminus A|X) \geq \Delta(B \setminus A|Y)$$

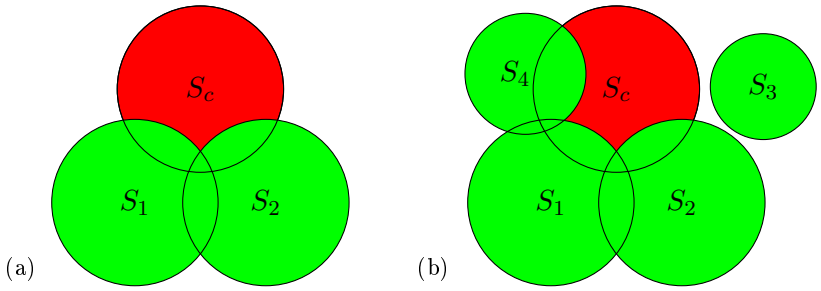
$$f((B \setminus A) \cup (A \cap B)) - f(A \cap B) \geq f((B \setminus A) \cup A) - f(A)$$

$$f(B) + f(A) \geq f(A \cup B) + f(A \cap B).$$

□

Gondoljuk most a diszkrét deriváltat haszonnak. A szubmodularitás definíciója által sugallt szemlélet szerint egy adott e elem hozzávétele egy egyre bővülő halmazsorozathoz nemnövekvő haszonnal jár, azaz a szubmoduláris függvények kielégítenek egy természetes tulajdonságot, amit a csökkenő hasznok elvének nevezhetünk. Ennek észben tartása gyakran hasznunkra lehet a későbbiekben.

A vízhálózatos példánkhoz készített ábra szemléletesen is mutatja a csökkenő hasznokat. Legyen itt S_i az a terület, ahol történt szennyezéseket az $s_i \in V$ helyre telepített érzékelő gyorsan érzékeli. Ha s_1 és s_2 mellett telepítjük s_3-at és s_4-et, a több előre telepített érzékelő hatókörnyezetébe S_e nagyobb helyen metsz bele, azaz csökken a haszna: $\Delta(s_e|\{s_1, s_2\}) \geq \Delta(s_e|\{s_1, s_2, s_3, s_4\})$.



1.1. ábra. Ugyanazon érzékelő hozzávétele egy szűkebb szenzorhalmazhoz (a) legalább akkora hasznot hoz, mint egy bővebb halmazhoz való hozzáadása (b).

Néhány gyakran előforduló példa szubmoduláris függvényekre:

1.1.5. Definíció. *$f : 2^V \to \mathbb{R}$ moduláris, ha minden $A, B \subseteq V$ esetén $f(A \cap B) + f(A \cup B) = f(A) + f(B)$.*

1.1.6. Példa. A moduláris függvények szubmodulárisak 1.1.4 alapján. Hasonlítanak a lineáris függvényekre, mivel a diszkrét deriváltjaik konstansok: $\Delta(e|B) = \Delta(e|A)$ minden A, B és $e \notin A \cup B$ esetén. Ezért egy f moduláris függvény, feltéve, hogy $f(\emptyset) = 0$, felírható $f(S) = \sum_{e \in S} w(e)$ alakban, ahol $w : V \to \mathbb{R}$ megfelelő súlyfüggvény.

1.1.7. Példa. Halmazok be- és kifok-függvénye szubmoduláris. Könnyen ellenőrizhető az 1.1.4-es állítás segítségével.

1.1.8. Definíció. (Monotonitás) *Egy $f : 2^V \to \mathbb{R}$ függvény monoton, ha minden $A \subseteq B \subseteq V$ esetén $f(A) \leq f(B)$.*

1.1.9. Megjegyzés. Egy f függvény pontosan akkor monoton, ha a diszkrét deriváltjai nemnegatívak, azaz ha $A \subseteq V$ és $e \in V$ esetén $\Delta(e|A) \geq 0$. A monoton szubmoduláris függvények egy fontos részhalmaza az, amelyre fennáll, hogy minden $A \subseteq B \subseteq V$ és $e \in V$-re $\Delta(e|A) \geq \Delta(e|B)$. Ez a tulajdonság kissé különbözik a 1.1.3 definíciótól abban, hogy nem szükséges az, hogy $e \notin B$.

1.1.10. Példa. A súlyozott fedések szubmodulárisak. Legyen X egy halmaz, $g : 2^X \to \mathbb{R}_+$ moduláris függvény, $w : X \to \mathbb{R}$ a g-hez tartozó súlyfüggvény, V pedig X egy részhalmazcsaládja. Ekkor egy $S \subseteq V$ részcsalád esetén az $f(S) := g\Big(\bigcup_{v \in S} v\Big) = \sum_{x \in \bigcup_{v \in S} v} w(x)$ függvény monoton szubmoduláris. Mindkét állítás könnyen ellenőrizhető.

A vízvezetékes példában X vonatkozhat szennyezési eseményekre, $w(x)$ mérheti egy adott x esemény súlyosságát, és minden lehetséges $v \in V$ szenzorhelyhez hozzárendeljük az észlelt X-beli eseményeket.

1.1.11. Példa. Üzletláncok esetén tegyük fel, hogy egy $V = \{1, \ldots, n\}$ halmazból szeretnénk kiválasztani azokat a helyeket, ahol létesítményeket állítanánk fel adott m darab vevő kiszolgálására. A j helyen megnyitott létesítmény az i vevő számára $M_{i,j}$ értékben nyújt szolgáltatást , ahol $M \in \mathbb{R}^{m \times n}$. Amennyiben az összes vevő a számára legnagyobb értékű szolgáltatást nyújtó létesítményt választja, a vevőknek nyújtott szolgáltatások összértékét az $f(S) = \sum_{i=1}^{m} \max_{j \in S} M_{i,j}$ függvény szolgáltatja. Legyen itt $f(\emptyset) = 0$. Ekkor ha minden i és j esetén $M_{i,j} \geq 0$, akkor könnyen belátható, hogy $f(S)$ monoton szubmoduláris.

Ez a modell más alkalmazásokban is használható. Például az érzékelő-elhelyezési feladatunkban $M_{i,j}$ vonatkozhat a j szenzor által az i esetben hozott haszonra, ahol a hasznosságot például a szennyezés észlelési idejének várható csökkentésében mérhetjük.

Most nézzük a szubmoduláris függvények néhány hasznos tulajdonságát.

1.1.12. Állítás. *Szubmoduláris függvények nemnegatív lineáris kombinációja szubmoduláris. Más szóval ha $g_1, ..., g_n : 2^V \to \mathbb{R}$ szubmoduláris és $\alpha_1, ..., \alpha_n \geq 0$, akkor $f(S) := \sum_{k=1}^{n} \alpha_k g_k(S)$ szubmoduláris.*

Bizonyítás. A definíció miatt szubmoduláris függvények nemnegatív számszorosa és összege is szubmoduláris. □

Az előbbi tulajdonság szerint bonyolult szubmoduláris célfüggvényeket felírhatunk egyszerűbb szubmoduláris alkotóelemek összegeként.

1.1.13. Állítás. *Minden f szubmoduláris függvény és $A, B \in V$ halmazok esetén*

$$f(A) + \Delta(B|A) \leq f(A) + \sum_{e \in B} \Delta(e|A).$$

Bizonyítás. 1.1.2 és 1.1.3 felhasználásával adódik. □

Nézzük, mi a kapcsolat a szubmodularitás és a konkavitás között:

1.1.14. Állítás. *$g : \{0, ..., |V|\} \to \mathbb{R}$ esetén $f(S) := g(|S|)$ szubmoduláris $\Leftrightarrow$ g konkáv.*

Bizonyítás. Ha g konkáv, az azt jelenti, hogy minden $S \subseteq T \subseteq V, e \in V$-re $|S| \leq |T|$ miatt $\Delta_f(e|S) \geq \Delta_f(e|T)$, ami épp a szubmodularitás definíciója. Visszafelé, legyen f szubmoduláris. Ekkor legyen $S_0 \subset S_1 \subset ... \subset S_{|V|} = V$. Ekkor minden $0 < k < |V|, e \notin S_k$-ra

$$\Delta_f(e|S_k) \leq \Delta_f(e|S_{k-1})$$

$$g(k+1) - g(k) \leq g(k) - g(k-1)$$

$$g(k+1) + g(k-1) \leq 2g(k),$$

azaz g konkáv. □

1.1.15. Állítás. *A monoton szubmodularitás megőrződik csonkoláskor: ha* $f : 2^V \to \mathbb{R}$ *monoton szubmoduláris, akkor* $g(S) := min\{f(S), c\}$ *is az, bármely* $c \in \mathbb{R}$ *számra .*

Bizonyítás. Könnyen adódik a diszkrét deriváltas definícióból. □

Most nézzünk példát arra, hogy míg a csonkolás megőrzi a szubmodularitást, két szubmoduláris függvény minimuma, maximuma és különbsége nem feltétlen szubmoduláris.

1.1.16. Példa. (Kivonás) Legyen $V = \{1, 2\}$. Az első oszlopban a függvények neve, az első sorban a behelyettesítési értékek olvashatók. Itt míg f és g monoton szubmoduláris, $f - g$ nem szubmoduláris, mert $\Delta_{f-g}(2|\emptyset) < \Delta_{f-g}(2|\{1\})$.

	$\emptyset$	{1}	{2}	{1,2}
f	0	1	1	2
g	0	1	2	2
$f - g$	0	0	-1	0

1.1.17. Példa. (Minimumvétel) Ugyanazokkal a jelölésekkel f és g szubmoduláris, de $h := \min(f, g)$ nem az, mert $h(\emptyset) + h(1, 2) > h(1) + h(2)$.

	$\emptyset$	{1}	{2}	{1,2}
f	0	1	-1	0
g	0	-1	1	0
$\min(f, g)$	0	-1	-1	0

1.1.18. Példa. (Maximumvétel) Legyen V={1,2,3}, a jelösések az előbbiek. Itt f és g szubmoduláris, de $h := \max(f, g)$ nem az, mert $h(1) + h(1, 2, 3) > h(1, 2) + h(2, 3)$.

	$\emptyset$	{1}	{2}	{3}	{1,2}	{1,3}	{2,3}	{1,2,3}
f	0	0	0	1	0	1	1	1
g	0	1	0	1	0	1	0	0
$\max(f, g)$	0	1	0	1	0	1	1	1

Míg a szubmoduláris függvények egyes tulajdonságai a konkáv függvényekére emlékeztetnek, úgy egyes tulajdonságuk a konvex függvényekkel hozza őket kapcsolatba. Ilyen például az, hogy a konvex függvényekhez hasonlóan, melyeket hatékonyan lehet minimalizálni, a szubmodulárisok minimalizálása lehetséges (erősen) polinomiális időben [1].

1.2. Szubmoduláris függvények offline maximalizálása

Legyen $U \subseteq 2^V$ halmazok családja, U elemeit hívjuk megengedetteknek. Szeretnénk megoldani a $\max_{S \in U} f(S)$ feladatot. Az erre a célra írt algoritmusok vizsgálatakor felmerül a kérdés, hogy egy $S \subseteq V$ esetén hogy kapjuk meg $f(S)$-et. Feltesszük, hogy adott egy orákulum, mely minden S-re azonnal megadja az $f(S)$ értéket. A legegyszerűbb példa a megoldandó feladatra a számosságkorlátos, ahol $U = \{S \big| \, |S| \leq k\}$ valamely k-ra. Már ez is NP-nehéz a szubmoduláris függvények több osztályára nézve [1]. Ebből adódóan érdemes tárgyalni az optimumot elméletileg is biztosan hatékonyan közelítő algoritmusokat.

1.2.1. Példa. (Mohó algoritmus) A $\max_{S \in U} f(S)$ közelítésére monoton szubmoduláris függvényeknél számosságkorlátos esetben kézenfekvő lehetőség a mohó algoritmus, mely egy üres S_0 halmazból indul, és az i. lépésben hozzávesz egy e elemet, melyre maximális a $\Delta(e|S_{i-1})$, azaz

$$S_i = S_{i-1} \cup \{\arg\max_e \Delta(e|S_{i-1})\}.$$

A következő tétel bizonyítja, hogy a mohó algoritmus közelítést ad az optimális megoldásra.

1.2.2. Tétel. (Nemhauser, Wolsey, Fisher 1978) *[1] Legyen $f : 2^V \to \mathbb{R}_+$ nemnegatív monoton szubmoduláris függvény, és legyenek $\{S_i\}_{i \geq 0}$ a mohó algoritmus által választott halmazok. Ekkor minden $k, l \in \mathbb{N}_+$-ra*

$$f(S_l) \geq (1 - e^{-l/k}) \max_{S:|S| \leq k} f(S).$$

Speciálisan, $l = k$-ra $f(S_k) \geq (1 - 1/e) \max_{S:|S| \leq k} f(S)$.

Bizonyítás. Rögzítsük le l-et és k-t. Legyen $S^* \in \arg\max\{f(S) : |S| \leq k\}$. A monotonitás miatt az általánosság megszorítása nélkül feltehetjük, hogy $|S^*| = k$. Legyen S^* egy tetszőleges sorbarendezése a következő: $\{v_1^*, \ldots, v_k^*\}$. Ekkor minden $i < l$-re áll az alábbi egyenlőtlenséglánc:

$$f(S^*) \leq f(S^* \cup S_i) \tag{1.3}$$

$$= f(S_i) + \Delta(S^*|S_i) \tag{1.4}$$

$$\leq f(S_i) + \sum_{v \in S^*} \Delta(v|S_i) \tag{1.5}$$

$$\leq f(S_i) + \sum_{i=1}^{k} (f(S_{i+1}) - f(S_i)) \tag{1.6}$$

$$\leq f(S_i) + k(f(S_{i+1}) - f(S_i)) \tag{1.7}$$

(1.3) f monotonitásából ered, (1.4) 1.1.2 miatt igaz, (1.5) 1.1.13-ből következik, (1.6) azért áll, mert S_{i+1} mohón származott S_i-ből, (1.7) pedig azt használja ki, hogy $|S^*| \leq k$. Tehát

$$f(S^*) - f(S_i) \leq k(f(S_{i+1}) - f(S_i)) \tag{1.8}$$

Legyen $\delta_i := f(S^*) - f(S_i)$, így (1.8)-et újraírhatjuk $\delta_i \leq k(\delta_i - \delta_{i+1})$ alakban, ezt átrendezve kapjuk:

$$\delta_{i+1} \leq \left(1 - \frac{1}{k}\right)\delta_i. \tag{1.9}$$

Ezért $\delta_l \leq \left(1 - \frac{1}{k}\right)^l \delta_0$. Jegyezzük meg, hogy $\delta_0 = f(S^*) - f(\emptyset) \leq f(S^*)$, mert $f \geq 0$. Ezekből a jól ismert $1 - x \leq e^{-x}$, minden $x \in \mathbb{R}$-re egyenlőtlenséggel adódik, hogy

$$\delta_l \leq \left(1 - \frac{1}{k}\right)^l \delta_0 \leq e^{-l/k} f(S^*). \tag{1.10}$$

δ_l helyére $f(S^*) - f(S_l)$-et helyettesítve, majd az egyenletet átrendezve kapjuk a keresett $f(S_l) \geq (1 - e^{-l/k}) f(S^*)$ korlátot.

□

Bár a tételt eredetileg csak az $l = k$ esetre bizonyították, az $l \neq k$ eset ismerete hasznos. Például ha a szennyezési feladatban a mohó algoritmusnak megengedjük, hogy k helyett $5k$ érzékelőt válasszon ki, akkor a k elem optimumának közelítési hányadosa ≈ 0.63-ról ≈ 0.99-re nő.

Nemhauser és Wolsey ugyancsak 1978-ban bebizonyította, hogy nem létezik olyan polinomiális sok részhalmazt megvizsgáló algoritmus, mely az $(1 - 1/e)$-közelítésnél jobbat tudna.

2. fejezet

Online algoritmusok

2.1. Determinisztikus online algoritmusok

Az algoritmusok hagyományos módon való tervezésekor feltesszük, hogy az algoritmus birtokába kerül az összes bemeneti adatnak, még mielőtt a kimeneti adatokat meg kéne adja nekünk. Sok valós alkalmazásban ez a feltételezés nem állja meg a helyét, ilyenkor az algoritmusunk még az előtt el kell kezdje a kimenet generálását, mielőtt a teljes bemenetet ismerné.

Az általunk tárgyalt online feladatok felfoghatók kérelem-válasz-játékként.

2.1.1. Definíció. (Kérelem-válasz-játék) *A kérelem-válasz-játék egy R kérelemhalmazból, egy véges A válaszhalmazból, és az $f_n : R^n \times A^n \to \mathbb{R} \cup \{\infty\}, n \in \mathbb{N}$ költségfüggvényekből áll. Minden n-re valamely $F_n \subseteq A^n$ halmazt hívjunk a megengedett válaszok halmazának. Legyen $f := \bigcup_{n \in \mathbb{N}} f_n$.*

2.1.2. Definíció. *Egy n hosszú $\underline{r}$ kérelemsorozat **költségét** jelölje $c(\underline{r}) := \min\{f_n(\underline{r}, \underline{a}) | \underline{a} \in F_n\}$. Ha az $\underline{r} \in R^n$ kérelem- és $\underline{a} \in F_n$ válaszsorozatok esetén $f_n(\underline{r}, \underline{a}) = c(\underline{r})$, akkor az $\underline{a}$ válaszsorozatot **optimálisnak** nevezzük az $\underline{r}$-re.*

Először tekintsük a determinisztikus online algoritmusokat, azaz az olyanokat, melyek futásuk során nem használnak véletleneket.

2.1.3. Definíció. (Determinisztikus online algoritmus) *Egy G determinisztikus online algoritmus egy $g_i : R^i \to A, i \in \{1, 2, \dots\}$ alakú függvényekből álló sorozat. Egy $\underline{r} = (r_1, \dots, r_n)$ kérelemsorozat esetén definiáljuk a $G(\underline{r}) = (a_1, \dots, a_n) \in F_n$ válaszsorozatot, ahol $a_i = g_i(r_1, \dots r_i), i \in \{1, \dots n\}$. Legyen G költsége $\underline{r}$-en $c_G(\underline{r}) = f_n(\underline{r}, G(\underline{r}))$.*

Online feladatok és online algoritmusok környezetében a jobb megkülönböztethetőség érdekében a hagyományos feladatok és algoritmusok elnevezését néha kiegészítjük az offline jelzővel.

Az online algoritmusok hatékonyságát az összehasonlító-analízis segítségével mérjük, ahol az online algoritmus által adott kimenetet hozzámérjük az optimális kimenethez.

2.1.4. Definíció. (Determinisztikus online algoritmus c-közelítősége) *Legyen* $\alpha : \mathbb{R} \to \mathbb{R}$ *lineáris függvény. Azt mondjuk, hogy egy* G *determinisztikus online algoritmus* α*-közelítő, ha bármely* $\underline{r}$ *kérelemsorozatra* $c_G(\underline{r}) \leq \alpha(c(\underline{r}))$ *teljesül. Amennyiben* $\alpha(x) = dx+e$*, ahol* $d > 0$*, azt mondjuk, hogy a közelítési hányados* d*.*

2.1.5. Megjegyzés. Amennyiben egy G algoritmus α-közelítő, és $\alpha(x) = dx + e$, esetenként a G algoritmust d-közelítőnek mondjuk. Hasonló átfogalmazásokat lehet eszközölni az α-közelítőségek későbbi definícióinál is.

Néhány példa online feladatra:

2.1.6. Példa. Feladatelosztás. Feladatok egy sorát kell beosztani gépek egy halmazán, egy adott célfüggvényt optimalizálva (például az utolsónak befejezett feladat befejezési időpontját minimalizálva). A feladatok egyenként érkeznek, és azonnal hozzá kell rendelni őket egy géphez, a későbbi feladatok ismerete nélkül.

Az utolsó befejezési időpont minimalizálását kérelem-válasz-játékként a következőképp fogalmazhatjuk meg. Legyen M a gépek, R pedig a lehetséges feladatok időigényének halmaza. Legyen $A = M$, azaz a válasz azt határozza meg, melyik géphez rendeljük a feladatunkat. Ekkor $f_n = \max\limits_{m \in M} \sum\limits_{a_i = m} r_i$.

2.1.7. Példa. Adatszerkezetek. Szeretnénk egy adatszerkezetet (pl. egyszeresen láncolt listát, fát) karbantartani úgy, hogy minél kisebb költséggel hozzáférjünk a kért adatelemhez. A jövőbeli kéréseket nem ismerjük előre.

Lista-karbantartás. Legyen a lista hossza k és $R = \{R_1, \ldots, R_k\}$ a lista elemeiből álló kérelemhalmaz. Legyen $L = \{[a, b] \big| a$ és b a lista elemeinek egy-egy permutációja$\}$, azaz L a lista lehetséges megváltoztatásainak halmaza. Legyen $M \subseteq L$ a megengedett megváltoztatások halmaza. Az $A = \{1, \ldots, k\} \times M$ válaszhalmaz elemeinek első koordinátái azt mondják meg, hogy hányadik helyen érhető el a kért elem, a második koordináta pedig hogy az adott állapotból milyen sorrendbe rendezi át az algoritmusunk a listát az aktuális lépésben. Legyen $f_n(\underline{r}, \underline{a}) = \sum\limits_{l=1}^{k} a_{l,1}$, azaz az n. kérésig tett lépések száma az adatelemek elérésekor.

A fa karbantartásának leírása hasonló a listáéhoz. A különbség az A válaszhalmaz leírásában van. A lista esetéből kiindulva legyen M a fa megengedett megváltoztatásainak halmaza. Legyen N az összes lehetséges elérési útvonal halmaza. $A = \{1, \ldots, k\} \times M \times N$, $a_{i,1}$-ben tároljuk a kért elem szintjét a fában. Így a költségfüggvények szintén $f_n(\underline{r}, \underline{a}) = \sum_{l=1}^{k} a_{l,1}$ módon alakulnak.

2.1.8. Példa. Memória-kezelés, azaz a számítógépek memóriájának kezelése, ahol a memória egy kicsi gyors, és egy nagy lassú részből áll. Konkrétabb példaként, mialatt a felhasználó önkényesen lapozgat, az a célunk, hogy a gyakran látogatott oldalak a gyors memóriában legyenek tárolva, minél kevesebbszer kelljen új oldalt betölteni oda.

A lapozási feladat leírásához képzeljük azt, hogy a k oldalt tárolni tudó gyorsmemória helyezkedik el egy lista első k helyén, a lassú memória pedig mögötte. Az összes kezelt oldal száma legyen l. Ekkor $R = \{R_1, \ldots, R_l\}$ az oldalakból mint a lista elemeiből áll. M elemei a helybenhagyás és az olyan transzpozíciók, ahol a kicserélt elemek közül pontosan egy van a gyorsmemóriában. Legyen $A = \{1, \ldots, l\} \times M$. A válaszok első koordinátáiban tároljuk a kért elem helyét. Ekkor $f_n(\underline{r}, \underline{a}) := \big|\{a_{i,1} | a_{i,1} > k\}\big|$.

2.1.9. Példa. Szindbád esete a háremhölgyekkel. Mivel Szindbád nem ismeri előre a szerre elhaladó hölgyek szépségét, a legszebb hölgy kiválasztása számára online feladat. (A kiválasztott háremhölgy szépsége várható értékének maximalizálása szintén.)

Egy lehetséges megfogalmazás a kérelem-válasz-játékok nyelvén. Tegyük fel, hogy a háremhölgyek száma ismert, n . Közülük az i-edik a legszebb ($i \in \{1, \ldots, n\}$). Jelölje r_j azt a kérdést, hogy óhajtja-e Szindbád a j. hölgyet. A j. hölgy megjelenésekor az első j hölgy szépsége között ismert egy rendezés, legyen ez bekódolva az r_j kérésbe. Legyen $A = \{1, 0\}$, azaz a lehetséges válaszok az igen és a nem. Az egyszerűség kedvéért tegyük fel, hogy az $\underline{r}$ vektor n koordinátából áll, azaz minden esetben az összes háremhölgy elvonul Szindbád előtt. Legyen a k hosszú megengedett válaszsorozatok halmaza $F_k = \{a \big| \, a \in A^k, \langle a, a \rangle \leq 1\}$. Mivel egyértelmű, hogy Szindbád sikerrel jár vagy kudarcot vall, definiáljuk a költségfüggvényeket a következőképp. Legyen f_m azonosan nulla minden $m < n$ esetén,

$$f_n(\underline{r}, \underline{a}) = \begin{cases} 0 & \text{ha } a_i = 1 \text{ és } a \text{ megengedett,} \\ 1 & \text{különben.} \end{cases}$$

2.2. Példa: lapozási feladat

A következőkben közelebbről vizsgálunk egy online példát.

2.2.1. Példa. (Lapozási feladat) Vegyük a memóriáról szóló 2.1.8 példát. Tegyük fel, hogy a gyors memóriaegység egyszerre k oldalt tud tárolni, a lassú kapacitása pedig bármilyen nagy lehet. A kérelmek egy-egy oldal megjelenítésére szólnak. Ha az oldal a gyors memóriában található, a kérelmet kiszolgáljuk, ha pedig a lassúban van, akkor hibát jelzünk. Ebben az esetben a gyors memóriából át kell helyezni egy oldalt a lassúba, hogy a megürült helyre betölthessük a kért oldalt. Egy lapozó-algoritmus eldönti, hogy hiba esetén melyik oldalt vegyük ki a gyors memóriából. A minimalizálandó költség a jelzett hibák száma. A lapozási feladatot oldja meg a következő két determinisztikus online algoritmus.

2.2.2. Definíció. *A lapozási feladatra adott LRU (legkevésbé használt, Least Recently Used) determinisztikus online algoritmus hibajelzéskor azt az oldalt veszi ki a gyors memóriájából, amelyik legutóbbi kérése a legrégebb történt.*

2.2.3. Definíció. *A lapozási feladatra adott $FIFO$(First-In First Out) determinisztikus online algoritmus hibajelzéskor a gyorsmemóriában legrégebb óta bent levő oldalt veszi ki.*

2.2.4. Definíció. *A lapozási feladatra adott MIN offline algoritmus hibajelzéskor azt az oldalt veszi ki a gyorsmemóriából, amelyik kérése a jövőben legmesszebb fog történni.*

2.2.5. Tétel. *[5] A MIN optimális.* □

Legyen n_A az A algoritmus gyors memóriájában tárolható oldalak száma. Jelölje $F_A(s)$ az A algoritmus által az s bemeneten elkövetett hibák számát.

2.2.6. Tétel. *Legyen A egy lapozási feladatra adott tetszőleges determinisztikus online algoritmus. Feltéve, hogy $n_A \geq n_{MIN}$, létezik tetszőlegesen hosszú s kérelemsorozat úgy, hogy*

$$F_A(s) \geq (n_A/(n_A - n_{MIN} + 1))F_{MIN}(s).$$

Bizonyítás. Alkotunk egy olyan n_A hosszú kérelemsorozatot, melyen A n_A hibát vét, míg, a MIN csak $(n_A - n_{MIN} + 1)$-et. Az első $(n_A - n_{MIN} + 1)$ kérés érkezzen olyan odalakra, melyek sem az A, sem a MIN gyors memóriájában nincsenek benne, így ebben a fázisban mindkét algoritmus $(n_A - n_{MIN} + 1)$ hibát vét. Legyen S azon oldalak halmaza, melyek eredetileg benne voltak a MIN memóriájában, vagy szerepeltek az első $(n_A - n_{MIN} + 1)$ kérés között. Ekkor $|S| = n_A + 1$, azaz létezik olyan eleme, ami pillanatnyilag nincs A gyors memóriájában. A következő $n_{MIN} - 1$ kérés érkezzen olyan S-beli oldalakra, melyek

pillanatnyilag nincsenek A gyors memóriájában. Ezáltal míg A a kérelemsorozat második felében mind az $n_{MIN} - 1$ esetben hibát vét, a MIN a leírásából adódóan sosem, hiszen memóriájában tartja az utolsó $n_{MIN} - 1$ kért oldalt. Összegezve, mialatt eme n_A hosszú kérelemsorozat alatt az A végig hibázik, a MIN $(n_A - n_{MIN} + 1)$ hibát vét. Az előbbi eljárást kellően sokszor ismételve kapjuk a tétel állítását. □

2.2.7. Megjegyzés. A bizonyításból látszik, hogy ha $n_A < n_{MIN}$, akkor tetszőlegesen hosszú kérelemsorozat készíthető úgy, hogy az A mindig hibázzon, a MIN pedig sose.

2.2.8. Következmény. Az előző tételbe $k := n_A = n_{MIN}$-t írva azt kapjuk, hogy a lapozási feladat megoldására $c < k$ esetén nem létezik c-közelítő determinisztikus online algoritmus.

2.2.9. Tétel. *Bármely s kérelemsorozatra*

$$F_{LRU}(s) \leq (n_{LRU}/(n_{LRU} - n_{MIN} + 1))F_{MIN}(s) + n_{MIN}.$$

Bizonyítás. Az első kérelem után bizosan van az LRU és a MIN memóriájában közös oldal, mégpedig a legutóbb kért. Legyen t s egy olyan részsorozata, mely nem tartalmazza s első elemét, és amely folyamán az LRU $f \leq n_{LRU}$-szor hibázik. Jelölje p a közvetlen t előtt kért oldalt. Ha t alatt az LRU kétszer is hibázik ugyanazon oldal kérésekor, akkor t-nek tartalmazni kell kérést legalább $n_{LRU} + 1$ különböző oldalra. Ugyanez igaz, ha p-n hibázik t alatt. Ha az előbbi két eset közül egyik sem következik be, akkor $f \leq n_{LRU}$ miatt a MIN t folyamán legalább $f - n_{MIN} + 1$-szer hibázik.

Osszuk s-et fel $s_0, s_2, \ldots, s_k$-ra, úgy, hogy s_0 tartalmazza az első kérést és legfennebb n_{LRU} hibát kövessen el rajta az LRU, $i \in \{1, \ldots, k\}$ esetén pedig s_i-n pontosan n_{LRU} hibát ejtsen az LRU. Ekkor $s_1, \ldots, s_k$ mindegyikén az LRU és a MIN hibáinak aránya legfennebb $n_{LRU}/(n_{LRU} - n_{MIN} + 1)$. Ekkor s_0 alatt amennyiben az LRU f_0-szor hibázik, a MIN legalább $f_0 - n_{MIN}$-szer ejt hibát. Mindent összevetve kapjuk a tételt. □

2.2.10. Megjegyzés. A tételben szereplő additív tag abból adódik, hogy kezdetben a MIN és az LRU gyors memóriájában teljesen különböző oldalak szerepelhetnek. Ezt a tagot eltüntethetjük azáltal, hogy feltesszük, hogy kezdetben az LRU gyors memóriájában megtalálható a MIN gyors memóriájának összes eleme, és ezeknél frissebben nem volt használva egyetlen más elem sem.

2.2.11. Tétel. *Amennyiben $n_{LRU} = n_{MIN} =: k$, az LRU k-közelítő.*

Bizonyítás. Felhasználva 2.2.8-t, és 2.2.9 $n_{LRU} = n_{MIN}$ esetét, a tételt kapjuk. □

2.2.12. Tétel. *Amennyiben $n_{FIFO} = n_{MIN} =: k$, a FIFO k-közelítő.* □

2.3. Véletlen online algoritmusok

Egy természetesen felmerülő kérdés, hogy érhetünk-e el jobb eredményt nemdeterminisztikus algoritmusokat használva. Ennek megválaszolásához definiálni kell a randomizált online algoritmusok c-közelítőségét.

2.3.1. Definíció. (Véletlen (randomizált, véletlenített) online algoritmus) *Legyen D determinisztikus online algoritmusok egy halmaza, P egy eloszlás D-n, ahol minden $d \in D$ esetén $P(d)$ annak az esélye, hogy d fut le. Ekkor a $G(D, P)$ párost véletlen online algoritmusnak hívjuk.*

2.3.2. Megjegyzés. Az egyszerűség kedvéért egy $G(D, P)$ véletlen online algoritmust gyakran G-vel jelölünk. Bármely $\underline{r}$ kérelemsorozatra a $G(\underline{r})$ válaszsorozat, így a $c_G(\underline{r})$ költség is valószínűségi változó.

2.3.3. Állítás. *Legyen $G(D, P)$ egy véletlen online algoritmus. Ekkor* $\mathbf{E}[c_{G(D,P)}(\underline{r})] = \sum_{d \in D} P(d) c_d(\underline{r})$. □

2.3.4. Definíció. *Egy G véletlen online algoritmust valamely α lineáris függvény esetén α-közelítőnek hívunk, ha bármely $\underline{r}$ bemeneti sorozatra $\mathbf{E}[c_G(\underline{r})] \leq \alpha(c(\underline{r}))$.*

Mivel a c-közelítőség szempontjából fontos, hogy a vizsgált algoritmusunk minden $\underline{r}$ kérelemsorozaton jól teljesítsen, gondolhatunk arra, hogy a bemenetet egy ellenség szolgáltatja. A következő típusú ellenfeleket szokás vizsgálni:

2.3.5. Definíció. (Feledékeny ellenfél) *A feledékeny ellenfél (oblivious adversary) legenerálja a teljes $\underline{r}$ kérelemsorozatot, még mielőtt az online algoritmus kielégített volna egyetlen kérelmet is. A feledékeny ellenfél ismeri az általa generált kérelemsorozat optimális megoldását, így az ő költsége $c(\underline{r})$.*

Az alkalmazkodó offline ellenfél (adaptive offline adversary) megfigyelheti az online algoritmust, és a következő kérelmet adhatja az eddigi kérelmekre kapott válaszok alapján. A kérelemsorozatot optimális módon teljesíti.

2.3.6. Definíció. (Alkalmazkodó offline ellenfél) *Egy Q alkalmazkodó offline ellenfél egy $q_k : A^k \to R \cup \{stop\}$ függvénysorozat, ahol $d_Q \in \mathbb{N}$, $k \in \{1, \ldots, d_Q\}$ és q_{d_Q} csak a stop értéket veheti fel. Egy G determinisztikus online algoritmus és Q alkalmazkodó ellenfél esetén legyen $\underline{r}(G, Q) = (r_1, \ldots, r_n)$ a tényleges kérelemsorozat, $\underline{a}(G, Q) = (a_1, \ldots, a_n)$ a tényleges válaszsorozat, $n \leq d_Q$ ezek hossza. Ezen objektumokat a következőkben leírtak szerint, egyenként számoljuk ki, a következő sorrendben: $r_1, a_1, r_2, a_2, \ldots, r_n, a_n, n$.*

Legyen $r_{i+1} := q_i(a_1, \ldots, a_i), i \in \{0, \ldots, n-1\}$, $\underline{a}(G, Q) := G(\underline{r}(G, Q))$ *és* $q_n(\underline{a}(G, Q)) =$ *stop.*

A G *algoritmus költsége a* Q *ellenfél esetén legyen* $c_G(Q) := f_n(\underline{r}(G, Q), \underline{a}(G, Q))$. *A* Q *ellenfél költsége a* G *algoritmus esetén* $c_Q(G) = c(\underline{r}(G, Q))$.

Egy $H(D, P)$ *véletlen online algoritmus esetén* $\underline{r}(H(D, P), Q)$, $\underline{a}(H(D, P), Q)$ *és* $n = n(H(D, P), Q)$ *valószínűségi változók. Ekkor a* $H(D, P)$ *algoritmus várható költsége a* Q *ellenfél esetén* $\mathbf{E}[c_{H(D,P)}(Q)] = \sum_{d \in D} P(d) c_d(Q)$. *A* Q *ellenfél várható költsége a* $H(D, P)$ *algoritmus esetén* $\mathbf{E}[c_Q(H(D, P))] = \sum_{d \in D} P(d) c_Q(d)$.

Az alkalmazkodó online ellenfél az alkalmazkodó offline ellenfélhez hasonló, azzal a különbséggel, hogy minden kérelmet online kell teljesítsen, azaz nem ismeri az algoritmus által a jelenben és jövőben adott (véletlen) válaszokat.

2.3.7. Definíció. (Alkalmazkodó online ellenfél) *Egy* $S = (Q, P)$ *alkalmazkodó online ellenfél egy alkalmazkodó offline ellenfél, ellátva egy* $P = \{p_1, \ldots, p_n\}$ *függvénysorozattal, ahol* $p_k : A^k \to A, k \in \{1, \ldots, d_Q\}$. *Legyen* G *egy determinisztikus online algoritmus. Mivel* $\underline{r}(G, S)$ *független* P*-től,* $\underline{r}(G, S) = \underline{r}(G, Q)$, $\underline{a}(G, S) = \underline{a}(G, Q)$ *és* $c_G(S) = c_G(Q)$. *Az* S *alkalmazkodó online ellenfél válaszsorozata* $\underline{b}(G, S) = (b_1, \ldots, b_n)$, *ahol* $n := n(G, Q)$ *és* $b_{i+1} = p_i(a_1, \ldots, a_i), i \in \{0, \ldots, n-1\}$. *Legyen* S *költsége* G *esetén* $c_S(G) = f_n(\underline{r}(G, S), \underline{b}(G, S))$.

Az előbbi definícióhoz hasonlóan egy S *véletlen online algoritmus esetén* $\mathbf{E}[c_S(H(D, P))] = \sum_{d \in D} P(d) c_S(d)$.

2.3.8. Megjegyzés. Determinisztikus online algoritmust tekinthetünk speciális randomizált online algoritmusnak. Figyeljük meg, hogy ebben az esetben a három ellenfélfogalom egybeesik: mindhárom annyit tehet, hogy megválasztja a bemenetet. Ezért a 2.1.4 definícióból adódóan determinisztikus online algoritmusnál a c-közelítőség szempontjából mindegy, hogy használjuk-e bármely ellenfelet, vagy sem.

2.3.9. Definíció. (α-közelítőség feledékeny ellenfelekhez) *Egy* G *véletlen online algoritmus* α*-közelítősége bármely feledékeny ellenfélhez jelentse ugyanazt, mint a 2.3.4-ben leírt* α*-közelítősége.*

2.3.10. Definíció. (α-közelítőség alkalmazkodó offline ellenfelekhez) *Egy* G *véletlen online algoritmust* α*-közelítőnek nevezünk bármely alkalmazkodó offline ellenfélhez, ha bármely* Q *alkalmazkodó offline ellenfél esetén* $\mathbf{E}[c_G(Q)] \leq \mathbf{E}[\alpha(c_Q(G))]$.

2.3.11. Definíció. (α -közelítőség alkalmazkodó online ellenfelekhez) *Egy A randomizált online algoritmus α-közelítőnek nevezünk bármely alkalmazkodó online ellenfélhez, ha bármely S alkalmazkodó online ellenfél esetén $\mathbf{E}[c_G(S)] \leq \mathbf{E}[\alpha(c_S(G))]$.*

2.3.12. Megjegyzés. A költségminimalizáló feladatból adódik, hogy $\alpha(x) = ax + b$ esetén $a \geq 1$. Azonban amennyiben a feladatunk valamely célfüggvény maximalizálása, az α-közelítőségeket értelemszerűen újradefiniáljuk, ekkor $a \leq 1$ lesz. A rájuk vonatkozó, értelemszerűen átfogalmazott állítások igazak maradnak.

A következőkben rangsoroljuk az ellenfeleket.

2.3.13. Tétel. *[3] Ha létezik minden alkalmazkodó offline ellenfelet α-közelítő véletlen online algoritmus, akkor létezik minden alkalmazkodó offline ellenfelet α-közelítő determinisztikus online algoritmus is.* □

2.3.14. Következmény. Az alkalmazkodó offline ellenfelet ugyanannyira lehet közelíteni determinisztikus online algoritmussal, mint véletlen online algoritmussal. □

Az előbbi tétel üzenete az, hogy az alkalmazkodó offline ellenfél jobb közelítésében nem segít a randomizálás.

2.3.15. Állítás. *Legyen G egy véletlen online algoritmus. Ha G b-közelítő bármely alkalmazkodó online ellenfélhez, akkor b-közelítő bármely feledékeny ellenfélhez is.*

Bizonyítás. Minden feledékeny ellenfelet utánozni lehet alkalmazkodó online ellenféllel. □

2.3.16. Tétel. *[3] Legyen G egy véletlen online algoritmus. Ha G b-közelítő bármely alkalmazkodó online ellenfélhez, és c-közelítő bármely feledékeny ellenfélhez, akkor $(b \cdot c)$-közelítő bármely alkalmazkodó offline ellenfélhez.* □

2.3.17. Következmény. Legyen G egy véletlen online algoritmus. Ha G a-közelítő bármely alkalmazkodó offline ellenfélhez, b-közelítő bármely alkalmazkodó online ellenfélhez, és c-közelítő bármely feledékeny ellenfélhez, akkor $a \geq b \geq c$.

Bizonyítás. 2.3.15-ből és 2.3.16-ból következik. □

Mielőtt az eredményeket összefoglalnánk, az ellenfeles környezet mellett ejtsünk pár szót a sztochasztikus esetről is, amikor a kérelmek véletlenszerűen érkeznek.

2.3.18. Definíció. (Sztochasztikus kérelemsorozat) *Legyen n egy valószínűségi változó, mely a $\{0, \dots N\}$, $N \in \mathbb{N}$ értékeket veheti fel. Legyen $\underline{r}_n = (\rho_0, \dots, \rho_n)$ olyan valószínűségi változók egy vektora, a valószínűségi változók értékkészlete az R kérelemhalmaz egy részhalmaza. Ekkor az $\underline{r}_n$ kérelemsorozatot sztochasztikusnak nevezzük.*

2.3.19. Definíció. (sztochasztikus α-közelítőség) *Egy $G(D,P)$ algoritmust egy $\underline{r}_n$ sztochasztikus kérelemsorozat és α lineáris függvény esetén sztochasztikusan α-közelítőnek nevezünk (az optimumhoz), ha $\mathbf{E}[c_G(\underline{r}_n)] \leq \alpha(\mathbf{E}[c(\underline{r}_n)])$, ahol a várhatóértéket $\underline{r}_n$ és P szerint vesszük.*

2.3.20. Állítás. *Ha egy G online algoritmus α-közelítő minden feledékeny ellenfélhez, akkor sztochasztikusan is α-közelítő.*

Bizonyítás. Az, hogy a $G(D,P)$ online algoritmus α-közelítő minden feledékeny ellenfélhez, azzal ekvivalens, hogy bármely $\underline{r}$ bemeneti sorozatra $\mathbf{E}_P[c_G(\underline{r})] \leq \alpha(c(\underline{r}))$. Ebből következik, hogy bármely $\underline{r}_n$ sztochasztikus kérelemsorozat esetén $\mathbf{E}_{(\underline{r}_n,P)}[c_G(\underline{r})] \leq \alpha(\mathbf{E}_{(\underline{r}_n,P)}[c(\underline{r})])$, azaz G sztochasztikusan α-közelítő. □

2.3.21. Állítás. *Ha létezik sztochasztikusan α-közelítő $G(D,P)$ véletlen online algoritmus, akkor létezik sztochasztikusan α-közelítő H determinisztikus online algoritmus is.*

Bizonyítás. $G(D,P)$ sztochasztikus α-közelítősége a következőt jelenti:

$$\mathbf{E}_{(\underline{r}_n,P)}[c_G(\underline{r}_n)] \leq \alpha(\mathbf{E}_{(\underline{r}_n,P)}[c(\underline{r}_n)]),$$

ami a 2.3.3 állítás alapján az egyelőtlenség a következő alakba írható:

$$\mathbf{E}_{\underline{r}_n}\left[\sum_{d\in D} P(d)c_d(\underline{r})\right] \leq \alpha(\mathbf{E}_{\underline{r}_n}[c(\underline{r}_n)]).$$

A legutóbbi egyenlőtlenség csak akkor állhat fenn, ha létezik olyan $d \in D$ determinisztikus online algoritmus, amire $\mathbf{E}_{\underline{r}_n} c_d(r_n) \leq \alpha(\mathbf{E}_{\underline{r}_n}[c(\underline{r}_n)])$.

□

2.3.22. Következmény. Sztochasztikus esetben ugyanannyira lehet közelíteni az optimumot determinisztikus online algoritmussal, mint véletlen online algoritmussal. □

A fentebb bemutatott eredményeket az alábbi táblázatban tekinthetjük át, ahol az (i,j) helyen található szám az i típusú online algoritmussal bármely j típusú ellenfél ellen, illetve sztochasztikus esetben elérhető legjobb közelítési hányados. A determinisztikus eset összes ellenfeles mezőjét egyesítettem, mivel c-közelítési fogalmak ezekben egybeesnek. Megjegyezendő, hogy az ellenfél nélküli és feledékeny ellenfeles esetek definíció szerint ugyanazt jelentik, így az ellenfél nélküli eredményeket a feledékeny ellenfeles oszlopból lehet kiolvasni.

	Sztochasztikus	Feledékeny	Alkalmazkó online	Alkalmazkodó offline
Determinisztikus	d	a		
Véletlen	d	c	b	a

2.1. táblázat. Az online költségminimalizálási feladatban determinisztikus és véletlen online algoritmusokkal a különböző ellenfelekkel szemben, illetve a sztochasztikus esetben elérhető legalacsonyabb közelítési hányadosok.
Itt $d \leq c \leq b \leq a$ és $a \leq cb$.

2.4. Példa: maximális párosítás online keresése

Ebben az alfejezetben főként a [6] cikk tartalmát dolgozzuk fel.

Sok, a Szindbád-feladathoz köthető problémát meg lehet fogalmazni. Egy számunkra érdekes kérdés az online házasításé, azaz a páros gráfokban történő maximális párosítás keresése. Látni fogjuk, hogy itt különböző típusú ellenfelekhez más-más közelítési eredményeket érhetünk el. A feladat leírása a következő.

Legyen $H(U, V, E)$ egy $2n$ ponton értelmezett teljes párosítást tartalmazó páros gráf, ahol két pont között pontosan a kölcsönös szimpátia esetén van él. Legyen B egy $\{0,1\}^{n\times n}$ mátrix, melyben eltároljuk a H szerkezetét. A mátrix sorai tartozzanak az U-beli pontokhoz (fiúk), oszlopai a V-beli pontokhoz (lányok), $B_{i,j} := 1$, ha $\{U_i, V_j\} \in E$, különben 0. Szeretnénk minél nagyobb párosítást készíteni H-ban online módon. Tegyük fel, hogy a fiú pontok jelen vannak, a lány pontok pedig egy előre meghatározott sorrendben érkeznek, és a hozzájuk tartozó élek az érkezésükkor kerülnek birtokunkba. A feladatunk eldönteni minden érkező lány pontra, hogy melyik fiú ponthoz adjuk (ha adjuk) úgy, hogy minél nagyobb párosításhoz jussunk. Az ezzel ekvivalens feladatot megfogalmazhatjuk a B mátrixra is, ahol a oszlopok (melyek a lányokhoz tartoznak) a lányok érkezési sorrendjének függvényében szerre válnak ismertté. Legyen ez a sorrend $\sigma = B_{\sigma_1}, \ldots, B_{\sigma_n}$. Az általánosság megszorítása nélkül megváltoztathatjuk az oszlopok mátrixon belüli helyét. Ezentúl feltételezzük, hogy $\sigma = B_n, B_{n-1} \ldots, B_1$. Egy A randomizált online algoritmus esetén legyen $A(H)$ valószínűségi változó az A algoritmus által H bemeneten konstruált párosításának elemszáma.

2.4.1. Definíció. *Az online párosítási feladatban az előbbiekben leírt jelölések mellett az A randomizált online algoritmus teljesítményét a $p(A) := \min_H E[A(H)]$ módon értelmezzük, ahol a várhatóértéket az A által használt véletleneken vesszük.*

2.4.2. Állítás. *A párosítási feladatra létezik $1/2$-közelítő mohó determinisztikus online algoritmus, és nem létezik olyan determinisztikus online algoritmus, mely $(1/2+\epsilon)$-közelítő*

lenne (tetszőleges ellenfélhez), semely $\epsilon > 0$ esetén.

Bizonyítás. Egy mohó algoritmus, mely amennyiben lehet, mindig hozzáad egy lányt egy tetszőleges választható fiúhoz, minden esetben elér egy olyan párosításhoz, amelyik tovább nem bővíthető, azaz mérete legalább $\lceil n/2 \rceil$. Másrészt a program leírásának ismeretéből adódóan egy (tetszőleges típusú) ellenfél bármely determinisztikus online algoritmus által elért legnagyobb párosítás méretét korlátozni tudja $\lceil n/2 \rceil$-ben, például úgy, hogy az elsőnek megjelenő $\lceil n/2 \rceil$ oszlopba 1-eseket ír, a maradékban pedig csak azokba, amelyikeket a determinisztikus algoritmus az első $\lceil n/2 \rceil$ lépésben párosított. □

2.4.3. Megjegyzés. Az előbbi állítás és a 2.3.14 értelmében véletlen online algoritmussal minden alkalmazkodó offline ellenfelet lehet 1/2-közelíteni, és nem lehet$(1/2+\epsilon)$-közelíteni semmilyen $\epsilon > 0$ esetén.

2.4.4. Állítás. *Randomizált online angoritmusok által generált párosítások várható méretének egy alkalmazkodó online ellenfél $n/2 + O(\log n)$-es felső korlátot tud szabni. Azaz nem létezik alkalmazkodó online ellenfélhez $(1/2+\epsilon)$-közelítő randomizált online algoritmus, semmilyen $\epsilon > 0$ esetén.*

Bizonyítás. Az ellenfél konstruálja B-t a következő módon. Adott $i \in \{0, \ldots, \lceil n/2 \rceil\}$ esetén $B_{j,n-i} = 1$ pontosan akkor, ha a j. sor nem eleme sem az algoritmus, sem az ellenfél által eddig készített párosításnak sem; az ellenfél a saját teljes párosításához egy tetszőleges 1-est választ az aktuális $(n-i)$. oszlopból. $i \in \{\lceil n/2 \rceil + 1, \ldots, n\}$ esetén $B_{j,n-i} = 1$ pontosan akkor, ha a j. sor nem eleme az ellenfél által eddig készített párosításnak; az előbbi esethez hasonlóan az ellenfél a saját teljes párosításához egy tetszőleges 1-est választ az aktuális $(n-i)$. oszlopból.

Hogy ezen ellenség ellen randomizált online algoritmus nem érhet el $n/2 + O(\log n)$-nél nagyobb párosítást, a következők miatt igaz. Egyrészt minden nem mohó randomizált online algoritmus (azaz mely nem feltétlenül rendel hozzá fiút a lányhoz, mikor erre lehetőség van) lecserélhető olyan mohóra (mely minden lehetőségkor hozzárendel a lányhoz fiút), mely legalább ugyanolyan jól teljesít átlagosan. Másrészt, bármely A mohó algoritmusra legyen $T(A)$ azon sorok halmaza, melyeket párosított a $B_n, B_{n-1}, \ldots, B_{\lfloor n/2 \rfloor}$ oszlopok valamelyikével mind az algoritmus, mind az ellenfél. Ekkor $E\big[|T(A)|\big] = O(\log n)$, és az A által konstruált párosítás mérete nem haladja meg az $n/2 + |T(A)|$ számot. □

Az alkalmazkodó ellenfél vizsgálata után nézzük, mit mondhatunk feledékeny ellenfél esetén.

2.4.5. Definíció. *A RANKING egy, az online párosításra adott algoritmus, mely két fázisból áll:*

Inicializálás: A fiú pontokat rendezzük véletlenszerűen, nevezzük ezt a sorrendet rangnak.

Párosítás: Ahogy megérkezik egy lány pont, rendeljük a legmagasabb rangú elérhető fiú ponthoz (amennyiben létezik ilyen).

2.4.6. Megjegyzés. Elég természetesnek tűnik egy még egyszerűbb algoritmust tanulmányozni, ami az érkező lány pontot egy tetszőleges elérhető fiú ponthoz rendeli, ez azonban a teljes felsőháromszög-mátrixon várhatóan $n/2+O(\log n)$ méretű párosítást fog produkálni. Ehhez képest a $RANKING$ éppen a fiú pontok rangsorolása miatt másként viselkedik, implicit módon előnyben részesítve azokat a fiúkat, melyek kevésbé voltak választhatók a múltban.

2.4.7. Tétel. *[7] A B mátrixhoz tartozó teljes párosítást tartalmazó $2n$ pontú gráfban, a $RANKING$ által adott párosítás várható mérete legalább $(1-1/e)+o(n)$. Tehát a $RANKING$ $(1-1/e)$-közelíti bármelyik feledékeny ellenfelet.* □

2.4.8. Tétel. *[7] $\epsilon > 0$ esetén nem létezik minden feledékeny ellenfelet $(1-1/e+\epsilon)$-közelítő online maximális párosítást kereső algoritmus.* □

Az előbbiek alapján felvázolhatjuk a következő táblázatot. Ahol a fogalmak egybeesnek, a mezők egyesítettek.

	Feledékeny	Alkalmazkó online	Alkalmazkodó offline
Determinisztikus	1/2		
Véletlen	$1-1/e$	1/2	1/2

2.2. táblázat. Jelen feladatban determinisztikus és véletlen online algoritmusokkal a különböző ellenfelekkel szemben elérhető legmagasabb közelítési hányadosok.

Mint látható, bár a véletlen online algoritmusoknak elvileg lehetőségük van magasabb közelítési hányadost elérni a determinisztikusoknál, az alkalmazkodó online ellenfélnél tapasztalt $O(\log n)$-es többlet még nem jelentkezik a közelítőség mértékében. Így a determinisztikus és véletlen algoritmusok hatékonysága jelen feladatban a feledékeny ellenfeles és ellenfél nélküli esetben különbözik lényegesen.

2.4.9. Megjegyzés. Az online maximális párosítás keresésének feladata felfogható szubmoduláris függvények összegének online maximalizálásaként is, így felfoghatjuk a következő fejezetben tárgyalt probléma egy sajátos eseteként.

Képzeljük ugyanis az eddigi $H(U,V,E)$ páros gráfot úgy, hogy egy árverésen U elemei az ügynökök, V elemei pedig a tárgyak, és minden $i \in U$ ügynök a $j \in N(i)$ tárgyak

közül pontosan egy darab megszerzésében érdekelt, azaz az i ügynök hasznossági függvénye $w_i(S) := \min\{S \cap N(i)\}$ alakban írható. Könnyen ellenőrizhető, hogy minden $i \in U$-ra w_i szubmoduláris. Így ebben a megfogalmazásban a feladatunk az ügynökök összhasznának maximalizálása.

3. fejezet

Online szubmoduláris árverések

3.1. Online szubmoduláris árverések

A 2.4 részben tárgyalt feladat vizsgálata indította el a tudományos munkák azon sorozatát, amibe illeszkedik az alább meghatározott online szubmoduláris kombinatorikus árverési feladat is. Gyakorlati alkalmazásai az árveréseken és online hirdetési felületeken kívül számos helyen elképzelhetők.

3.1.1. Definíció. (Online szubmoduláris kombinatorikus árverési feladat) *Tekintsünk egy árverést, ahol $|T| = t$ darab, előre ismeretlen tárgy érkezik egyenként, amiket érkezésük pillanatában hozzá kell rendeljünk az u darab ügynök valamelyikéhez. Azt, hogy tárgyak egy halmaza milyen hasznot hoz az i. ügynöknek, a $w_i : 2^T \to \mathbb{R}_+$ alakú, haszonfüggvénynek nevezett szubmoduláris hozzárendelés adja meg. Feladatunk a $\sum_{i=1}^{u} w_i(S_i)$ kifejezés maximalizálása, ahol S_i az i. ügynökhöz rendelt tárgyak halmaza, azaz $S_1, \ldots, S_u$ S egy partícióját alkotja.*

3.1.2. Megjegyzés. A maximalizálandó kifejezés azt méri, hogy az ügynökök összességének mennyire hasznos a tárgyak elosztása. Ez indokolhatja az árverési feladat angol szakirodalomban gyakran használatos *welfare maximization* elnevezést.

Feltesszük, hogy mindenki együttműködik (szeretné a közjót növelni), így nem foglalkozunk a feladat játékelméleti vonatkozásaival.

Akárcsak az 1 fejezetben, itt is feltesszük, hogy a szubmoduláris függvények értékeit egy orákulum kérésünkre azonnal szolgáltatja.

3.2. Ellenféllel szemben

Ebben az alfejezetben célunk olyan algoritmus keresése, amely minden, a futása előtt rögzített bemeneten jól teljesít. Látni fogjuk, hogy a mohó algoritmusok sok esetben az árverési feladatokban is hatékonynak bizonyulnak.

3.2.1. Definíció. (Mohó algoritmus) *A mohó algoritmus egy árverési feladatban az érkező tárgyat a legkisebb sorszámú olyan ügynökhöz rendeli, amelynek legnagyobb haszna származik belőle.*

3.2.2. Tétel. [8] *A mohó algoritmus online szubmoduláris árverési feladatokban* $1/2$*-közelítő, amennyiben a haszonfüggvények monotonok.*

Bizonyítás. Jelölje Q az eredeti feladatunkat, ahol t tárgy érkezik u ügynök részére. Tegyük fel, hogy az első tárgyat a mohó algoritmus a j. ügynökhöz rendelte. Jelöljük Q'-tel azt a feladatot, amit a Q-ból kapunk azáltal, hogy az érkező tárgyak sorának elejéről elhagyjuk az első (1 nevű) tárgyat, és a j. ügynök w_j haszonfüggvényét lecseréljük a $w'_j(S) := \Delta_{w_j}(S|\{1\})$ függvényre. Legyen $MOHÓ(Q)$ a mohó algoritmus futása során keletkezett összhasznát az ügynököknek, $OPT(Q)$ pedig az optimális összhasznot. Legyen $p = w_j(\{1\})$. Q' meghatározásából adódóan $MOHÓ(Q) = MOHÓ(Q') + p$. A következőkben megmutatjuk, hogy $OPT(Q) \leq OPT(Q') + 2p$.

Legyen $S_1, \ldots, S_u$ egy kiválasztott optimális elosztás, ahol S_i az i. ügynökhöz rendelt tárgyak halmazát jelöli. Tegyük fel, hogy az első tárgy eleme S_k-nak, azaz az optimális hozzárendelésben az első tárgy a k-adik ügynöknél van. Legyen S'_i az optimális elosztásból az első elem elhagyásával keletkezett elosztás Ekkor $S'_i = S_i$, minden $i \neq k$ esetén, és $S'_1, \ldots, S'_k$ egy lehetséges optimális megoldása Q'-nek. Hasonlítsuk össze $OPT(Q)$ és $OPT(Q')$ értékét. A k. ügynök kivételével mindegyikhez ugyanazokat a tárgyakat rendeljük mindkét esetben, a j. ügynök kivételével pedig mindegyiknek megegyezik a haszonfüggvénye. Az általánosság megszorítása nélkül feltehető, hogy $k \neq j$. A Q feladatról a Q'-re való áttérés során a k. ügynök legfennebb $w_k(\{1\})$ hasznot veszít, mivel a w_k szubmoduláris, továbbá a mohó algoritmus futása miatt $w_k(\{1\}) \leq w_j(\{1\}) = p$, így a k. ügynök legfennebb p értéket veszít. A j. ügynök is legfennebb p hasznot veszít, mivel w_j monotonitásából adódóan $w'_j(S_j) = w_j(S_j \cup \{1\}) - w_j(\{1\})) \geq w_j(S_j) - p$. Így $OPT(Q') \geq OPT(Q) - 2p$. A Q' feladatban szereplő haszonfüggvények szintén szubmodulárisak lesznek, így indukció segítségével nyer bizonyítást:

$$OPT(Q) \leq OPT(Q') \leq 2MOHÓ(Q') + 2p = 2MOHÓ(Q).$$

□

3.2.3. Megjegyzés. Könnyű belátni, hogy a mohó algoritmus legfennebb 1/2-közelítő, mivel a következő példán csak az optimum felét éri el:

	$\emptyset$	$\{1\}$	$\{2\}$	$\{1,2\}$
w_1	0	1	1	1
w_2	0	1	0	1

Amennyiben a haszonfüggvények nem monotonok, a mohó algoritmus esetenként nagyon rosszul teljesít. Egy példa erre:

	$\emptyset$	$\{1\}$	$\{2\}$	$\{1,2\}$
w_1	0	0	1	0
w_2	0	0	0	0

3.2.4. Megjegyzés. A mohó algoritmus polinomiális futásidejű.

A következőkben megmutatjuk, hogy amennyiben $NP \neq RP$, jelen feladatra nem létezik 1/2-nél nagyobb közelítési hányadosú algoritmus.

3.2.5. Definíció. (Fedés-kiértékelés) *Egy $w : 2^M \to \mathbb{R}_+$ függvényt fedés-kiértékelőnek (coverage valuation) hívunk, ha létezik egy olyan $\{A_j : j \in M\}$ halmazrendszer, amire $w(S) = |\bigcup_{j \in S} A_j|$, minden $S \subseteq M$ esetén.*

3.2.6. Állítás. *A fedés-kiértékelő függvények szubmodulárisak.* □

3.2.7. Definíció. (RP) *RP-belinek mondjuk az olyan feladatokat, melyekre léteztik olyan polinomiális futásidejű véletlen Turing-gép, mely NEM-mel tér vissza, ha a válasz NEM, és legalább 1/2 valószínűséggel IGEN-nel tér vissza, ha a válasz IGEN.*

3.2.8. Tétel. [7] *Ha $NP \neq RP$, $\delta > 0$ és fedés-kiértékelő haszonfüggvények esetén nem létezik olyan polinomiális futásidejű online algoritmus, mely $(1/2+\delta)$-közelítő lenne minden feledékeny ellenfélhez az online szubmoduláris árverési feladatban.* □

3.2.9. Következmény. Ha $NP \neq RP$ és $\delta > 0$ esetén nem létezik olyan polinomiális futásidejű online algoritmus, mely $(1/2 + \delta)$-közelítő lenne minden feledékeny ellenfélhez az online szubmoduláris árverési feladatban. □

3.2.10. Megjegyzés. Az eredményeket összegezve: a 3.2.2 tétel, a 3.2.9 következmény és az ellenfelek típusainak erősorrendje értelmében monoton szubmoduláris haszonfüggvények

esetén az árverési feladatra létezik 1/2-közelítő determinisztikus online algoritmus (pl. a mohó), és semmilyen $\delta > 0$ esetén nem létezik olyan polinomiális futásidejű véletlen online algoritmus, mely $(1/2 + \delta)$-közelítő lenne minden feledékeny, minden alkalmazkodó online, illetve minden alkalmazkodó offline ellenfélhez, hacsak NP nem egyenlő RP-vel.

3.2.1. Azonos tárgyak esetén

Egy természetesen felmerülő kérdés az, hogy mit mondhatunk akkor, ha az árverésen minden tárgy egyforma. Ekkor minden ügynök haszna egy, csak a hozzá rendelt tárgyak számától függő konkáv függvénnyé egyszerűsödik.

3.2.11. Tétel. [9] *Azonos érkező tárgyak esetén a mohó algoritmus az online szubmoduláris árverés egy optimális megoldását szolgáltatja.*

Bizonyítás. Minden $|A| = j$, $j \in \{1, \ldots, t\}$, $B \subset A$, $|B| = j - 1$ és minden $i \in \{1, \ldots u\}$ esetén legyen $a_{i,j} = \Delta_{w_i}(A|B)$, azaz az i. ügynöknek a j. tárgy által hozott haszon. Legyen S az összes ilyen $a_{i,j}$ szám multihalmaza. Tekintsük azt a MAX nevű feladatot, amikor ki szeretnénk választani t darabot az S elemei közül úgy, hogy ezek összege maximális legyen. Ezt megtehetjük mohó eljárással: minden lépésben az S-ben levő elemek közül kivesszük a legnagyobbat (ha több ilyen van, akkor közülük az első, majd a második index szerinti legkisebb sorszámút). Könnyen látszik, hogy ennek a feladatnak az optimuma felső korlátja az árverési feladatunknak.

Figyeljük meg, hogy w_i szubmodularitása miatt $a_{i,j} \geq a_{i,k}$, minden $k > j$ esetén. Legyen az $S_i = \{a_{i,j} | k \in \{1, \ldots, t\}\}$ multihalmaz. Tegyük fel, hogy a MAX feladatra adott mohó algoritmus a legutóbbi lépésben $a_{i,k}$-t választotta. Ekkor minden i-re ha az S_i multihalmazból a következőkben választ új elemet az algoritmus, akkor az szükségképpen az $a_{i,k+1}$ lesz, mivel a mohó eljárás leírása és $a_{i,k}$ k szerinti monoton csökkenősége miatt sem kisebb, sem nagyobb indexűt nem választhat az algoritmus. Így a MAX feladat megoldása megengedett megoldása lesz az eredeti árverési feladatunknak is.

Megjegyezendő, hogy egy ellenfél mozgástere annyiban merül ki, hogy megválaszthatja az érkező tárgyak számát, ez azonban nem változtat az optimalitáson. □

3.2.12. Megjegyzés. Figyeljük meg, hogy jelen esetben az optimalitáshoz nem szükséges, hogy a haszonfüggvények monotonok legyenek.

3.2.2. Azonos haszonfüggvények esetén

Az ellenféllel szembeni feladatnak egy másik természetesen adódó sajátos esete az, amikor az ügynökök haszonfüggvényei megegyeznek. Bár az erre a feladatra adott online algorit-

musok közelítési hányadosaira adhatók 1-nél kisebb felső korlátok, jelenleg nyitott kérdés, hogy e feladatban mohó algoritmus milyen hatékony, illetve hogy a leghatékonyabb-e.

3.2.13. Állítás. *Azonos haszonfüggvényű online monoton szubmoduláris árverési feladatokban a mohó algoritmus nem lehet* $(3/4+\delta)$*-közelítő, semmilyen* $\delta > 0$ *esetén.*

Bizonyítás.

Az állítás igazolásához elég mutatni egy olyan esetet, melyben a mohó algoritmus az optimum 3/4-ét éri el. Egy példa erre a következő. Két ügynök részére érkezik négy tárgy. Legyen az ügynökök halmaza $U = \{u_1, u_2\}$, Legyen az érkező tárgyak halmaza $T = \{A, B, C, D\}$, a tárgyak érkezzenek a névsor szerint.

Jelölje $w : 2^T \to \{0, 1, 2\}$ az ügynökök egymásával megegyező haszonfüggvényét.

Az alábbi táblázatokban a felső sor celláin belül egymás mellé írt betűk jelentsék azoknak a tárgyanak a neveit, melyek halmazára alkalmazzuk a w függvényt.

Minden $t \in T$ esetén legyen $w(\{t\}) = 1$.

	AB	AC	AD	BC	BD	CD	ABC	ABD	ACD	BCD	$ABCD$
w	1	2	2	2	1	2	2	2	2	2	2

A mohó algoritmus által adott hozzárendelések és összhasznaik körről-körre:

MOHÓ	1. kör	2. kör	3. kör	4. kör
Összhaszon :	1	2	3	3
u_1	A	A	AC	ACD
u_2		B	B	B

Egy optimális megoldás:

OPT	1. kör	2. kör	3. kör	4. kör
Összhaszon :	1	2	3	4
u_1	A	A	A	BD
u_2		B	BC	BC

Látható, hogy a mohó algoritmus által elért összhaszon 3, míg az optimum 4. Tehát jelen feladatban a mohó algoritmus az optimum 3/4-ét éri el.

□

3.2.14. Megjegyzés. Amennyiben a haszonfüggvény nem monoton, a mohó algoritmus esetenként nagyon rosszul teljesít. Egy példa erre:

	$\emptyset$	A	B	C	AB	AC	BC	ABC
w	0	0	ϵ	1	0	0	0	0

3.1. táblázat. Nem monoton közös haszonfüggvény esetén a mohó teljesíthet rosszul. Itt két ügynök és $\epsilon \searrow 0$ esetén az optimum nagyon kis részét éri el.

3.2.15. Állítás. *Azonos haszonfüggvényű online monoton szubmoduláris árverési feladatokban $\delta > 0$ esetén nem létezik minden alkalmazkodó offline ellenfelet $(3/4 + \delta)$-közelítő online algoritmus.*

Bizonyítás.

Vegyük a következő feladatot. Legyen az ügynökök halmaza $U = \{u_1, u_2\}$, az érkező tárgyak halmaza $T = \{A, B, C, D\}$, a tárgyak érkezzenek névsor szerint. Indirekt tegyük fel, hogy egy ALG nevű algoritmus valamely $\delta > 0$-ra $(3/4 + \delta)$-közelít minden alkalmazkodó offline ellenfelet. Minden $X \in T$ esetén legyen $w(\{X\}) = 1$, $w(\{A, B\}) = 1$. Ekkor az ALG algoritmus szükségszerűen nem ugyanahhoz az ügynökhöz rendeli az első kettőnek érkező A és B tárgyakat. Az általánosság megszorítása nélkül tegyük fel, hogy az A-t az első ügynökhöz rendeli. Legyen $w(\{A, C\}) = w(\{B, C\}) = 2$. Szintén az általánosság megszorítása nélkül tegyük fel, hogy az ALG a C-t az első ügynökhöz rendeli. Ekkor legyen $w(\{A, D\}) = w(\{A, C, D\}) = 2$, $w(\{B, D\}) = 1$. Végezetül a 3 és 4 elemű halmazok haszna legyen 2. Így a negyedik körben az algoritmus akármelyik ügynökhöz is rendeli az utolsó tárgyat, nem fog plusz hasznot hozni neki. Tegyük fel, hogy az első ügynökhöz rendeli. Megjegyezzük, hogy az így megadott w haszonfüggvény monoton szubmoduláris.

Az ALG által elért összhaszon 3, míg az optimum 4, így az ALG nem lehet $(3/4 + \delta)$-közelítő semmilyen $\delta > 0$-ra.

		A	B	C	D	AB	AC	AD	BC	BD	CD
w	0	1	1	1	1	1	2	2	2	1	2

3.2. táblázat. A 3 és 4 elemű halmazok haszna legyen 2.

Az ALG által adott hozzárendelések és összhasznaik körről-körre:

ALG	1. kör	2. kör	3. kör	4. kör
$Összhaszon$	1	2	3	3
u_1	A	A	AC	ACD
u_2		B	B	B

Egy optimális megoldás:

OPT	1. kör	2. kör	3. kör	4. kör
$Összhaszon$	1	2	3	4
u_1	A	A	A	AD
u_2		B	BC	BC

□

3.2.16. Állítás. *Azonos haszonfüggvényű online monoton szubmoduláris árverési feladatokban $\delta > 0$ esetén nem létezik minden alkalmazkodó online ellenfelet $(6/7 + \delta)$-közelítő online algoritmus.*

Bizonyítás. Tegyük fel indirekt, hogy valamely $\delta > 0$ esetén létezik egy minden alkalmazkodó offline ellenfelet $(6/7 + \delta)$-közelítő, ALG nevű online algoritmus.

A bizonyítás menete hasonló az előző állításéhoz, annyiban van hátrányban az alkalmazkodó online ellenfél az alkalmazkodó offline-nal szemben, hogy a harmadik tárgy érkezésekor nem tudhatja, hogy az ALG melyik ügynökhöz is rendeli az új tárgyat. Ekkor azt teheti az ellenfél, hogy ő ahhoz az ügynökhöz rendeli a harmadik tárgyat, amelyikhez az ALG legfennebb $1/2$ eséllyel rendelte.

Amennyiben az ALG és az ellenfél ugyanahhoz az ügynökhöz rendelte a harmadik tárgyat, úgy a negyedik tárgy már semmilyen esetben ne hozzon hasznot. Ekkor úgy az ALG, mint az ellenfél összhaszna 3 lesz.

Amennyiben az ALG és az ellenfél különböző ügynökökhöz rendelte a harmadik tárgyat, a negyedik lépésben az ellenfél tegyen úgy, mint az előző állítás bizonyításában. Ekkor az ALG összhaszna 3, míg az ellenfélé 4.

Így, ha az ellenfelet S-nek nevezzük, $\mathbf{E}[c_{ALG}(S)] = 3$, $\mathbf{E}[c_S(ALG)] \geq (3+4)/2$, így az ALG nem lehet $(6/7 + \delta)$-közelítő semmilyen $\delta > 0$-ra □

3.3. Független azonos eloszlású sztochasztikus bemenettel

3.3.1. Definíció. (Független azonos eloszlású sztochasztikus árverési feladat) *Legyen M az árverezendő tárgyak típusainak halmaza. Tegyük fel, hogy az árverésen K tárgy egy M feletti (ismert, vagy ismeretlen) D eloszlás szerint érkezik, egyenként. Legyen az ügynökök halmaza N. Feladatunk a D eloszlásból egyenként és függetlenül érkező tárgyak visszavonhatatlan társítása egy ügynökhöz.*

Ekkor a független azonos eloszlású sztochasztikus árverési feladatot egy $A(M, D, K, N)$ négyesként írhatjuk le, ahol célunk az ügynökök összhasznának maximalizálása.

A továbbiakban jelöljük a típusok számát m-mel, az ügynökök számát n-nel, az i. ügynök haszonfüggvényét pedig w_i-vel.

Az árverés előrehaladtával előfordulhat az, hogy egy ügynöknek egy adott típusú tárgyból több is birtokába kerül. Így az általa birtokolt tárgyak *halmaza* helyett a tárgyak *multihalmazáról* beszélhetünk, ugyanis a multihalmazokban egy elem többször is előfordulhat. Ezért szükségünk van a szubmodularitás multihalmazokra való kiterjesztésére. Egy lehetséges, az árverési feladatok szempontjából észszerű megoldást (a csökkenő hasznú függvények fogalmát) a következőkben definiálunk.

3.3.2. Megjegyzés. Legyen m_i az M típushalmaz i. eleme. Ekkor $\mathbb{N}^{|M|}$ és az M elemeiből álló multihalmazok között egy g bijekciót létesíthetünk úgy, hogy minden $x \in \mathbb{N}^{|M|}$ esetén $g(x)$ azzal az S multihalmazzal egyenlő, melyben az m_i típusú elemből x_i darab van. Így habár a következő definíció vektorokkal van megadva, egyértelműen átírható multihalmazos alakra.

3.3.3. Definíció. (Diszkrét derivált multihalmazok esetén) *Egy $f : \mathbb{N}^m \to \mathbb{R}$ függvény és $x \leq y$, $x, y \in \mathbb{N}^m$ esetén legyen $\Delta_f(y|x) := f(x+y) - f(x)$ az f-nek x-ben vett y irányú diszkrét deriváltja.*

3.3.4. Megjegyzés. Így a fenti definíció formálisan hasonló a 1.1.1 definícióhoz, azzal az értelmi különbséggel, hogy multihalmazok esetén az azonos típusú elemek számosságát is figyelembe kell venni.

3.3.5. Definíció. (Csökkenő hasznú függvények) *Egy $f : \mathbb{N}^m \to \mathbb{R}$ függvény csökkenő hasznú, ha minden $x \leq y$, $x, y \in \mathbb{N}^m$ és minden e_i, $i \in \mathbb{N}^m$ i. egységvektor esetén $\Delta_f(e_i|x) \geq \Delta_f(e_i|x)$.*

3.3.6. Megjegyzés. Egy f csökkenő hasznú függvényre az angol szakirodalomban azt mondják, hogy "f has the property of diminishing returns".

3.3.7. Megjegyzés. Egy $f : \mathbb{N}^m \to \mathbb{R}$ csökkenő hasznú függvény esetén $f|_{\{0,1\}^m}$ szubmoduláris.

3.3.8. Megjegyzés. Egy $f : \{0,1\}^m \to \mathbb{R}$ szubmoduláris függvény kiterjeszthető egy $f' : \mathbb{N}^m \to \mathbb{R}$ függvénnyé, például a következő módon: $f'(x) := f(\min(x, 1))$, ahol a minimumot koordinátánként vesszük.

3.3.9. Definíció. (Monotonitás) *Egy $f : \mathbb{N}^m \to \mathbb{R}$ függvény monoton, ha $f(x) \leq f(y)$, minden $x \leq y$ esetén.*

3.3.10. Megjegyzés. Egy $f : \mathbb{N}^m \to \mathbb{R}$ függvény pontosan akkor monoton, ha diszkrét deriváltjai nemnegatívak.

3.3.11. Definíció. (Mohó algoritmus) *Egy $A(M, D, K, N)$ feladatban hívjuk a következő algoritmust mohónak. A j tárgy érkezése előtt legyen az i. ügynökhöz rendelt tárgyak halmaza T_i, az ügynök haszonfüggvénye w_i (minden $i \in 1, \dots N$ esetén). A j tárgyat érkezésekor ahhoz az ügynökhöz rendeljük, mely haszonfüggvénye maximalizálja a $\Delta_{w_i}(j|T_i)$ értéket.*

3.3.12. Megjegyzés. A mohó algoritmusnak nincs szüksége D és K ismeretére.

A továbbiakban a mohó algoritmus teljesítményét vizsgáljuk.

3.3.13. Lemma. [7] *Egy $A(M, D, K, N)$ feladat várható optimuma, ahol a j elemhez a p_j valószínűség tartozik, S végigfut az összes legfennebb t elemű multihalmazon, $c_j(S) \geq 0$ pedig a j típusú elemek száma S-ben, felülről korlátozott az*

$$\begin{aligned} LP \quad = \quad & \max \sum_{i,S} x_{i,S} w_i(S) : \\ & \forall j : \sum_{i,S} x_{i,S} c_j(S) \leq p_j t; \\ & \forall i : \sum_{S} x_{i,S} = 1 \\ & \forall i, S : x_{i,S} \geq 0 \end{aligned}$$

kifejezéssel.

Bizonyítás. Tegyük fel, hogy az árverés során érkezett tárgyak multihalmaza T, $|T| = t$. Vegyük a feladat optimális (offline) megoldását, ennek összhasznát jelöljük $OPT(T)$-vel. Legyen $x_{i,S}$ annak a valószínűsége, hogy az optimális megoldásban az i. ügynökhöz rendelt multihalmaz az S. Ekkor az optimális összhaszon várhatóértéke $\mathbf{E}[OPT(T)] = \sum_{i,S} x_{i,S} w_i(S)$. A j típusú elemből minden S multihalmaz $c_j(S)$ darabot tartalmaz, így a hozzárendelt j típusú elemek várható száma $\sum_{i,S} x_{i,S} c_j(S)$. Ez az érték nem lehet több, mint a T-ben levő j típusú elemek várható száma, ami $\mathbf{E}[c_j(M)] = p_j t$. Így az $x_{i,S}$ értékek valóban a feladat egy megengedett optimális megoldását határozzák meg. □

Egy optimális megoldás összhasznának értékét jelöljük $OPT := \mathbf{E}[OPT(M)]$-mel.

3.3.14. Lemma. [7] *Tegyük fel, hogy a w_i haszonfüggvények monotonok és csökkenő hasznúak. Az árverés egy adott pontján legyen T_i az i ügynökhöz rendelt tárgyak pillanatnyi halmaza ($i \in \{1, \dots n\}$). Ekkor a következőnek érkező tárgy hozzárendeléséből származó várható haszon legalább $\frac{1}{t}(LP - \sum_i w_i(T_i))$.*

Bizonyítás. Legyen az $x_{i,S}$ értékekből alkotott vektor egy megengedett LP megoldás, és legyen $y_{ij} = \sum_S x_{i,S} c_j(S)$, ahol $c_j(S)$ az S-ben tárolt j típusú elemek száma. Az LP-megszorítások miatt $\sum_i y_{ij} \leq p_j t$. Használjuk a $T_i + S$ jelölést két multihalmaz uniójára. Ekkor a csökkenőhasznúság miatt

$$\Delta_{w_i}(S|T_i) \leq \sum_j c_j(S) \Delta_{w_i}(j|T_i).$$

A fenti alakú egyenlőtlenségeket $x_{i,S} \geq 0$-val megszorozva, majd i és S szerint összegezve kapjuk:

$$\sum_{i,S} x_{i,S} \Delta_{w_i}(S|T_i) \leq \sum_{i,j,S} x_{i,S} c_j(S) \Delta_{w_i}(j|T_i) = \sum_{i,j} y_{ij} \Delta_{w_i}(j|T_i).$$

Mivel a monotonitás miatt $\Delta_{w_i}(j|T_i) \geq 0$, definíció szerint $\Delta_{w_i}(S|T_i) = w_i(T_i + S) - w_i(T_i)$, az LP-megkötések miatt pedig $\sum_S x_{i,S} = 1$, az előbbi egyenlőtenségből kapjuk:

$$\sum_{i,S} x_{i,S} w_i(S) - \sum_i w_i(T_i) \leq \sum_{i,j} y_{ij} \Delta_{w_i}(j|T_i). \tag{3.1}$$

Most tekintsünk egy, a törtmegoldásokra vonatkozó hipotetikus hozzárendelési szabályt: ha j típusú elem érkezik, az i. ügynökhöz $\frac{y_{ij}}{p_j t}$ valószínűséggel rendeljük (az LP-megkötések miatt egy adott j-re e valószínűségek összege legfennebb 1). Mivel j típusú elem p_j valószínűséggel érkezik egy adott pillanatban, végsősoron egy körben az i. ügynökhöz j típusú elemet $\frac{y_{ij}}{t}$ valószínűséggel rendelünk. Így ezen véletlen hozzárendelési szabállyal

$$\mathbf{E}[\text{véletlen haszon}] = \sum_{i,j} \frac{y_{ij}}{t} \Delta_{w_i}(j|T_i). \tag{3.2}$$

Ekkor (3.1) és (3.2) alapján

$$\mathbf{E}[\text{véletlen haszon}] \geq \frac{1}{t} \Big(\sum_{i,S} x_{i,S} w_i(S) - \sum_i w_i(T_i) \Big).$$

Vegyük figyelembe, hogy a mohó algoritmus egy olyan ügynökhöz rendeli az éppen érkező tárgyat, amelyiknek maximális haszna származik ebből. Így bármely $x_{i,S}$ megengedett megoldás esetén a mohó algoritmus legalább akkora hasznot hoz, mint a véletlen hozzárendelési szabály, ezért az előbbiekből:

$$\mathbf{E}[\text{mohó haszon}] \geq \max \mathbf{E}[\text{véletlen haszon}] \geq \frac{1}{t}\Big(LP - \sum_i w_i(T_i)\Big).$$

□

3.3.15. Tétel. [7] *A mohó algoritmus $(1-1/e)$-közelítő a független azonos eloszlású sztochasztikus árverési feladatban, amennyiben az ügynökök haszonfüggvényei monotonok és csökkenő hasznúak.*

Bizonyítás. Jelölje az első τ elem érkezése utáni hozzárendeléseket $T_1^{(\tau)}, \ldots, T_n^{(\tau)}$. Ekkor a 3.3.13 és 3.3.14 lemma alapján a következőnek érkező tárgy hozzárendelése utáni összhaszon várhatóértéke

$$\mathbf{E}\Big[\sum_i w_i\big(T_i^{(\tau+1)}\big)\big|T_1^{(\tau)}, \ldots, T_n^{(\tau)}\Big] \geq \sum_i w_i\big(T_i^{(\tau)}\big) + \frac{1}{t}\Big(LP - \sum_i w_i\big(T_i^{(\tau)}\big)\Big).$$

A $(T_1^{(\tau)}, \ldots, T_n^{(\tau)})$ hozzárendelés szerint várhatóértéket véve a következőt kapjuk:

$$\mathbf{E}\Big[\sum_i w_i\big(T_i^{(\tau+1)}\big)\Big] \geq \mathbf{E}\Big[\sum_i w_i\big(T_i^{(\tau)}\big)\Big] + \frac{1}{t}\mathbf{E}\Big[LP - \sum_i w_i\big(T_i^{(\tau)}\big)\Big].$$

Vezessük be a $W(\tau) = \mathbf{E}\big[\sum_i w_i\big(T_i^{(\tau)}\big)\big]$ jelölést. Ekkor a legutóbbi egyenlőtlenség $W(\tau+1) \geq W(\tau) + \frac{1}{t}(LP - W(\tau))$, vagy ezzel ekvivalensen $LP - W(\tau+1) \leq (1 - \frac{1}{t})(LP - W(\tau))$ alakba írható. Indukcióval kapjuk:

$$LP - W(t) \leq \left(1 - \frac{1}{t}\right)^{\tau}(LP - W(0)) \leq e^{-\tau/t}LP$$

A mohó algoritmus által adott hozzárendelés várható összhaszna t tárgy érkezése után $W(t) = \mathbf{E}\big[\sum_i w_i(T_i^{(t)})\big]$. Az eredményeket összesítve kapjuk, hogy

$$W(t) \geq (1-1/e)LP \geq (1-1/e)OPT.$$

□

3.3.16. Tétel. [7] *Ha $NP \neq RP$, adott $\delta > 0$ és fedés-kiértékelő haszonfüggvények esetén nem létezik olyan polinomiális futásidejű online algoritmus, mely $(1-1/e+\delta)$-közelítő lenne független azonos eloszlású bemenettel az online szubmoduláris árverési feladatban.* □

3.3.17. Következmény. Ha $NP \neq RP$, adott $\delta > 0$ és csökkenő hasznú haszonfüggvények esetén nem létezik olyan polinomiális futásidejű online algoritmus, mely $(1-1/e+\delta)$-közelítő lenne független azonos eloszlású bemenettel az online szubmoduláris árverési feladatban. □

3.3.18. Megjegyzés. Az eredményeket összegezve: 3.3.15 tétel, a 3.3.17 következmény és az ellenfelek típusainak erősorrendje értelmében monoton csökkenő hasznú haszonfüggvények esetén az árverési feladatra létezik $(1-1/e)$-közelítő determinisztikus online algoritmus (pl. a mohó), és semmilyen $\delta > 0$ esetén nem létezik olyan polinomiális futásidejű véletlen online algoritmus, mely $(1-1/e+\delta)$-közelítő lenne minden feledékeny, minden alkalmazkodó online, illetve minden alkalmazkodó offline ellenfélhez, hacsak NP nem egyenlő RP-vel.

Most nézzük az előző feladat egy átfogalmazását:

3.3.19. Definíció. (Árverési feladat eloszlásból érkező független eloszlások esetén) *Legyen M az árverezendő tárgyak típusainak halmaza. Legyenek $d_1, \ldots, d_q$ M feletti eloszlások, D pedig egy $\{d_1, \ldots, d_q\}$ feletti eloszlás. Tegyük fel, hogy az árverésen K tárgy egy-egy D-ből érkező eloszlás szerint érkezik, egyenként. Legyen az ügynökök halmaza N. Feladatunk a D eloszlásból egyenként és függetlenül érkező tárgyak visszavonhatatlan társítása egy ügynökhöz.*

Ekkor az eloszlásból érkező független eloszlások esetén vett árverési feladatot egy $A(M, D, K, N)$ négyesként írhatjuk le, ahol célunk az ügynökök összhasznának maximalizálása.

3.3.20. Állítás. *A most definiált feladatra ugyanazok az állítások érvényesek, mint a 3.3.1-re, bizonyításaik lényegükben ugyanazok.* □

3.3.1. Azonos tárgyak esetén

3.3.21. Állítás. *A független azonos eloszlású sztochasztikus árverési feladatot, amennyiben azonos tárgyak érkeznek, a mohó algoritmus optimálisan megoldja.*

Bizonyítás. Nyilvánvaló, hogy azonos tárgyak esetén a feladat lényegében elveszíti a sztochasztikus jellegét, az egyetlen véletlen változó a kiválasztandó tárgyak száma. Így ugyanazt a feladatot kapjuk eredményül, mint amit a 3.2.1 részben tárgyaltunk. Ezért a 3.2.11 tétel alapján a mohó algoritmus optimális. □

3.4. Összefoglalás

A 3.2 és 3.3 alfejezetekben bemutatott eredmények alapján összeállíthatjuk a következő, az online szubmoduláris árverési feladat típusaira adható jelenleg ismert legjobb közelítéseket bemutató táblázatot. Mivel minden esetben a mohó az (ismert) legjobb algoritmus, ezért a táblázatban azt tüntettük fel, hogy a különböző esetekben mekkora a mohó algoritmus közelítési hányadosa, illetve hogy ismert-e, hogy (bizonyos feltevések mellett) nem létezik nála magasabb közelítési hányadost elérő algoritmus. Amennyiben tudjuk, ezt az „éles" szóval jelöljük, különben egy kérdőjellel. Amelyik esetben a mohó közelítési hányadosa nem ismert, ott az a legszűkebb intervallum szerepel, melyben mostani ismeretünk szerint nem tudjuk behatárolni az értékét. A táblázatban szereplő eredmények mellett szerepel, hogy honnan származnak.

Az alapfeladat az online szubmoduláris árverést jelenti monoton szubmoduláris (a sztochasztikus esetben pedig csökkenő hasznú) függvények esetén.

A táblázatban nem szereplő részeredmény az, hogy az azonos haszonfüggvények esetében nem létezik alkalmazkodó online ellenfelet 6/7-nél jobban közelítő online algoritmus.

	Sztochasztikus	Feledékeny	Alk. online	Alk. offline
Alapfeladat	$1 - 1/e$ *[7]*; éles, ha $NP \neq RP$ *[7]*	$1/2$ *[8]*; éles, ha $NP \neq RP$ *[7]*		
Azonos tárgyak	1 *[9]*; éles			
Azonos haszonfv.	$[1 - 1/e, 1]$,?	$[1/2, 3/4]$,?		

3.3. táblázat. A mohó algoritmus közelítési hányadosai és élessége az online monoton szubmoduláris árverési feladatban és sajátos eseteiben, a különböző típusú ellenfelek, ill. a sztochasztikus bemenet esetén.

Irodalomjegyzék

[1] A. Krause, D. Golovin, *Submodular Function Maximization*, 2012, 1-8, http://las.ethz.ch/files/krause12survey.pdf

[2] S. Albers, *Online Algorithms: a Survey*, 2003, 1-6, http://www.cc.gatech.edu/ mihail/D.7520reading/7520onlinesurvey.pdf

[3] S. Ben-David, A. Borodin, R. M. Karp, Tardos G., A. Wisgerson, *On the Power of Randomiation in Online Algorithms*, 1994, http://www.cs.technion.ac.il/ shai/BDBKTW.pdf

[4] D.D. Sleator, R. E. Tarjan *Amortized Efficiency of List Update and Paging Rules*, 1985, http://www.cs.cmu.edu/ sleator/papers/amortized-efficiency.pdf

[5] Bélády L. A., *A Study of Replacement Algorithms for Virtual Storage Computers*, 1966

[6] R. M. Karp, U. V. Vazirani, V. V. Vazirani *An Optimal Algorithm for On-line Bipartile Matching*, 1990, http://www.cs.berkeley.edu/ vazirani/pubs/online.pdf

[7] M. Karpalov, I. Post, J. Vondrák, *Online Submodular Welfare Maximization: Greedy is Optimal*, 2013, http://theory.stanford.edu/ jvondrak/data/online-alloc.pdf

[8] B. Lehmann, D. Lehmann, N. Nisan , *Combinatorial Suctions with Decreasing Marginal Utilities*, 2006, http://www.cs.huji.ac.il/ noam/submodular.pdf

[9] Magánértekezés Long Tran-Thanh-nal