19. AZ ÖSSZEHASONLÍTÁSOS RENDEZÉSEK MŰVELETIGÉNYÉNEK ALSÓ KORLÁTJAI

Ebben a fejezetben *aszimptotikus* (nagyságrendi) *alsó korlátot* adunk az összehasonlításokat használó rendező eljárások lépésszámára. Pontosabban, azt látjuk be, hogy egy n méretű input rendezése *nagyságrendben legalább* $n \log n$ *összehasonlítást* igényel. Ezt az alsó korlátot a *legrosszabb* és az *átlagos esetre* egyaránt bizonyítjuk. (Akik jártasabbak a valószínűségszámításban, szívesen mondanak *várható* lépésszámot az *átlagos* helyett.)

Az eredmény nemcsak az előző fejezetekben ismertetett hét rendezésre igaz, hanem az *összes* ismert (és még fel nem fedezett) összehasonlító rendező módszerre is érvényes.

Az 1. fejezetben láttuk, hogy az algoritmusok alapvető jellemzője a *műveletigényük*, vagyis az, hogy mennyire hatékonyan oldják meg a feladataikat. A *hatékonyságot* gyakorlati megközelítésben a futási idővel, elméletileg inkább a megtett lépések számával mérjük. A *lépésszám* általában egy vagy néhány meghatározó művelet végrehajtási számát jelenti.

A lépésszám olykor minden azonos méretű inputra ugyanannyi, és értéke pontosan megadható. A maximum-kiválasztás ismert eljárása minden n elemű tömbre $n-1$ összehasonlítást végez. A buborékrendezés bármely n méretű tömböt $n(n-1)/2$ összehasonlítással rendez.

Az algoritmusok lépésszáma azonban általában inputról-inputra változik, még ha azonos is a hosszuk. Ha tekintjük egy algoritmus összes azonos n elemszámú bemenő adatát, akkor a hatékonyságát a legnagyobb, illetve az átlagos lépésszámmal szoktuk jellemezni. Ezek az értékek lehetnek pontosan számolhatók, és lehet, hogy csak becsülni tudjuk azokat. A buborékrendezésről tudjuk, hogy a legkedvezőtlenebb input, a fordítva rendezett sorozat átrendezésére $n(n-1)/2$ cserét használ, míg az összes n hosszú bemenő sorozaton az átlagos (várható) csereszám ennek a fele: $n(n-1)/4$.

Ha egy algoritmus lépésszáma nem adható meg pontos képlettel az n inputméret függvényében, akkor nagyságrendben próbájuk becsülni. A kupacrendezés esetében nem lenne egyszerű egzakt módon megadni az elvégzett összehasonlítások vagy cserék maximális, illetve átlagos számát. Szerencsére az $(n \log n)$-es nagyságrendi becslés is tájékoztat alapvető módon az eljárás hatékonyságáról.

Ebben a fejezetben az összehasonlításos rendezéseket vesszük szemügyre. Nem egyedi hatékonyságukra irányul a vizsgálat, hanem általánosan érvényes alsó korlátot adunk műveletigényükre.

Viszonylag ritka „szerencsés” esetben meg lehet határozni egy feladatosztályhoz olyan alsó korlátot, amely érvényes minden megoldó eljárásra. A kiválasztásokról szóló 9-10. fejezetekben több alsó korlát elemzés is szerepel. Talán a legismertebb ezek közül az, hogy n elem közül a legnagyobbnak a kiválasztásához legalább $n-1$ összehasonlítás szükséges.

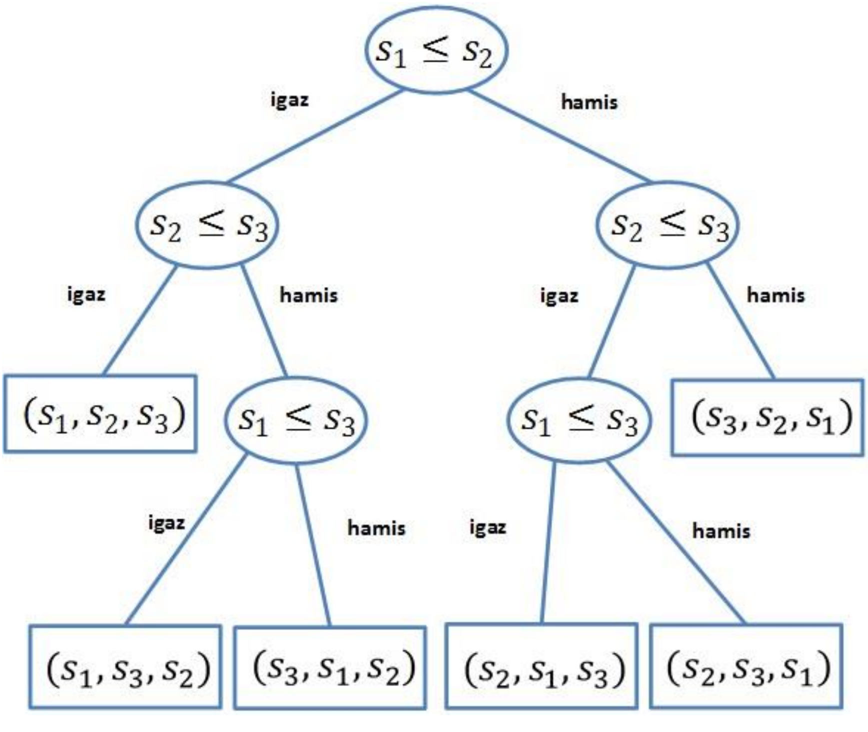
Visszatérve az összehasonlító rendezésekre, alsókorlátaik elemzéséhez bevezetünk egy szellemes elméleti adatstruktúrát, a döntési fát, amely egy algoritmus által feltett kérdések „lenyomata” minden bemenő adatra.

19.1. A döntési fa

A *döntési fa* elkészíthető minden *algoritmushoz* és annak adott n méretű *összes* bemenő adatához, feltéve, hogy ezek az adatok felsorolhatók és *számuk* meghatározható. A rendező eljárásokra ez teljesül, hiszen bemenő adatoknak tekinthetjük az $1, 2, \dots n$ számok permutációit, amelyek $n!$ száma közismert.

A döntési fa *belső pontjaiban* tartalmazza az algoritmus által feltett összes *igen*/*nem* kimenetelű *kérdést*, *minden* lehetséges n méretű bemenő adatra. (Az algoritmus ciklusai az adott n méretű bemenetre történő végrehajtás során egymás utáni lépések szekvenciájává „egyenesednek ki”, így iteratív vezérlési szerkezetet nem kell megjelenítenünk a fában.) Az esetleges értékadásokat sem helyezzük el a döntési fában. A kérdésekre adott válaszok *információtartalmát* – az értékadások adattranszformáló hatásának figyelembe vételével – minden bemenő adat útvonalán végig haladva fában összegyűjtjük, és a *levelekbe* írjuk. A döntési fában a levelek tartalma a *megoldást* fogalmazza meg az egyes inputokra.

A 19.1. ábrán látható $t_R(3)$ döntési fa egy olyan algoritmus működését szemlélteti, amelyet speciálisan *három elem rendezésére* terveztünk. Az R eljárás az s_1, s_2, s_3 elemek összehasonlítását végzi minden lehetséges input sorrendre. A kérdésekre adott válaszokból meghatározható az elemek nagyság szerint rendezett sorrendje. Ehhez kettő vagy három kérdés szükséges. (Ebben a kis eljárásban értékadások nem szerepelnek.)



19.1. ábra. Az $\boldsymbol{R}$ rendező algoritmus $\boldsymbol{t_R(3)}$ döntési fája 3 elemű sorozatokra

Ha pl. $s_1 = 2, s_2 = 3, s_3 = 1$, akkor az első kérdés *igaz* ágán jutunk el a második kérdéshez, amelyre *nem* a válasz (*hamis* ág), majd a harmadik kérdésnek ismét a *hamis* ágán jövünk ki. A levélben látható az elemek nagyság szerint rendezett (s_3, s_1, s_2) sorrendje.

Ha az elemek s_1, s_2, s_3 bemenő sorozata rendezett, vagy fordítva rendezett, akkor két összehasonlítás is elegendő a megoldáshoz. Láthatjuk tehát, hogy a döntési fa leveleinek magassága változó, a 2 és 3 értékeket veheti fel. A levelek száma pedig $3! = 6$, így a bemenő adatok minden sorrendjéhez tartozik az R algoritmusnak egy levélben végződő végrehajtási ága. A rendezések döntési fájának *magassága* és *leveleinek száma* között teremt összefüggést a következő állítás.

Lemma. Bármely R összehasonlító rendező eljárás $t_R(n)$ döntési fájának $h(t_R(n))$ magassága és az n elemszám között fennáll a következő összefüggés:

$$h(t_R(n)) \geq \log_2(n!)$$

Bizonyítás. A fa $h = h(t_R(n))$ magassága a gyökértől legtávolabbi levelek magasságával azonos. A bináris fában a h magasság szintjén 2^h elem befogadására van hely. A h magasság legalább akkora kell, hogy legyen, hogy helyt tudjon adni $n!$ számú levélnek, azaz teljesülnie kell a

$$2^h \geq n!$$

összefüggésnek. Ha mindkét oldal logaritmusát vesszük és visszaírjuk h eredeti jelentését, akkor a bizonyítandó összefüggést kapjuk. ■

Megjegyezzük, hogy példánkban teljesül a lemma állítása, hiszen fennáll a $8 = 2^3 \geq (3!) = 6$ összefüggés. Az R algoritmus a legkedvezőtlenebb inputjai éppen azok (a hatból négy ilyen), amelyek rendezésének lépései $h = 3$ magasságban található levelekben végződnek. Egy ilyen végrehajtási út során az eljárás három összehasonlítást végez. Az R algoritmus műveletigénye a legrosszabb esetben megegyezik a csúcstól legtávolibb levelek magasságával, vagyis a döntési fa magasságával. Ezt az összefüggést felhasználjuk a következő tétel bizonyításában.

19.2. Alsó korlát az összehasonlítások számára a legkedvezőtlenebb esetben

Az előző lemma alapján kimondhatjuk, és kevés számolás elvégzésével igazolhatjuk a következő állítást.

Tétel. Bármely R összehasonlításos rendező eljárás a legkedvezőtlenebb bemenő adata rendezése során nagyságrendben legalább $(n \log n)$ összehasonlítást végez, azaz

$$MÖ_R(n) = \Omega(n \log n)$$

Bizonyítás. A lemmához fűzött megjegyzés – a példa alapján – eljutott a következő általános érvényű összefüggésig. Egy R összehasonlító rendező algoritmus által végzett *összehasonlítások maximális száma* (legrosszabb eset) n méretű input esetén megegyezik az R-hez és n-hez tartozó *döntési fa magasságával*. Gondoljuk meg még egyszer, hogy a legtöbb összehasonlítás egy leghosszabb végrehajtási úton történik, amelynek hossza meghatározza a fa magasságát. Formálisan is kifejezve: $MÖ_R(n) = h(t_R(n))$. Összevetve ezt a lemmával:

$$MÖ_R(n) \geq \log_2(n!)$$

Alakítsuk át a jobb oldalon álló kifejezést:

$$\log_2(n!) = \log_2(n(n-1)\ldots 1) = \log_2 n + \log_2(n-1) + \cdots + \log_2 1 = \sum_{i=1}^{n} \log_2 i$$

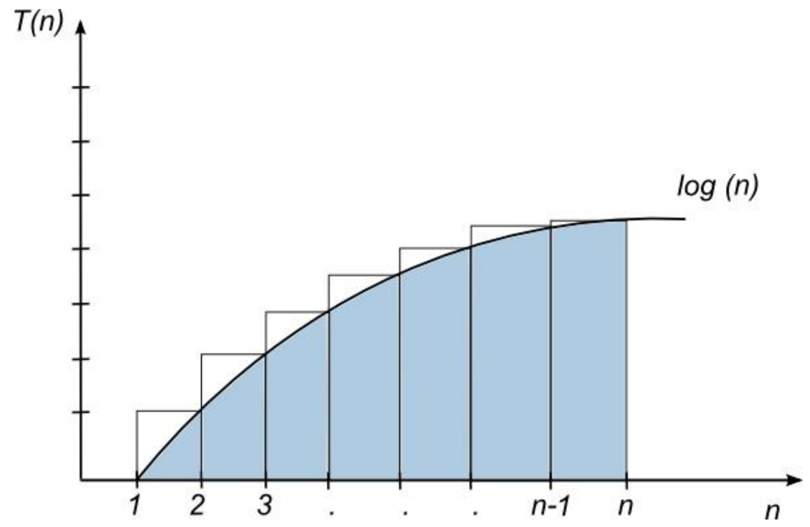
Tekintsük ezt az összeget, mint kívül írt téglalapok területének összegét, a logaritmus függvény integrál közelítő összegének és becsüljük alulról a határozott integrál értékével (lásd: 19.2. ábra).

$$\sum_{i=1}^{n} \log_2 i \geq \int_1^n \log_2 x \, dx = \log_2 e \int_1^n \ln x \, dx =$$

$$= \log_2 e \cdot [x \ln x - x]_1^n = \log_2 e \cdot n \ln n - \log_2 e \cdot n + \log_2 e =$$

$$= n \log_2 n - \log_2 e \cdot n + \log_2 e = \Omega(n \log n)$$

A következtetési láncolat elején és végét egybevetve, a bizonyítandó állítást kapjuk. ■



19.2. ábra. Összeg alsó becslése határozott integrállal

19.3. Alsó korlát az összehasonlítások számára átlagos esetben

Láttuk, hogy egy R rendező eljárás működését, amelyet az n méretű bemenő sorozatok rendezése során végez, a $t_R(n)$ döntési fa teljes körűen rögzíti. Egy adott sorozat rendezésnek megfelel egy olyan útvonal a fában, amely a gyökértől a megfelelő levélig terjed. Ezen az úton, a belső csúcsokban az algoritmus által végzett összehasonlítások szerepelnek. Az útvonalat lezáró levélben a bemenő sorozat rendezéséhez szükséges összes ismeret kerül elhelyezésre. A végrehajtott összehasonlítások száma megegyezik az adott útvonal hosszával, azaz a végén található levél magasságával.

A rendező eljárás (adott *n* méret melletti) átlagos összehasonlítási számához úgy jutunk, ha a döntési fa levélmagasságainak az átlagát képezzük. A levélmagasság összegre vezessük be az $lhsum(t)$ jelölést.

A döntési fákban minden belső pontnak két gyereke van, mivel a belső pontok *igen*/*nem* kimenetelű kérdéseket reprezentálnak. Nevezzük az ilyen alakú bináris fákat *tökéletesnek*. (Megjegyezzük, hogy egy *t* tökéletes fához *nem feltétlenül* tartozik olyan algoritmus, amelynek *t* a döntési fája lenne!)

Most megfogalmazunk egy olyan állítást, amely lehetőséget teremt az átlagos összehasonlítás-szám alsó becsléséhez.

Lemma. Az azonos számú levelet tartalmazó tökéletes fák közül levélmagasság összeg azokra a legkisebb, amelyek egyben *majdnem teljes* bináris fák.

Bizonyítás. Legyen *t* egy olyan *tökéletes fa*, amely *nem teljesíti* a majdnem teljesség követelményét, azaz levelei között *legalább két szintkülönbség* található. A 19.3. ábrán látható *t* tökéletes fában például az *A* és *B* levelek, valamint a *D* levél közötti szintkülönbség értéke kettő. Természetesen magasságértékeikre ugyanez mondható:

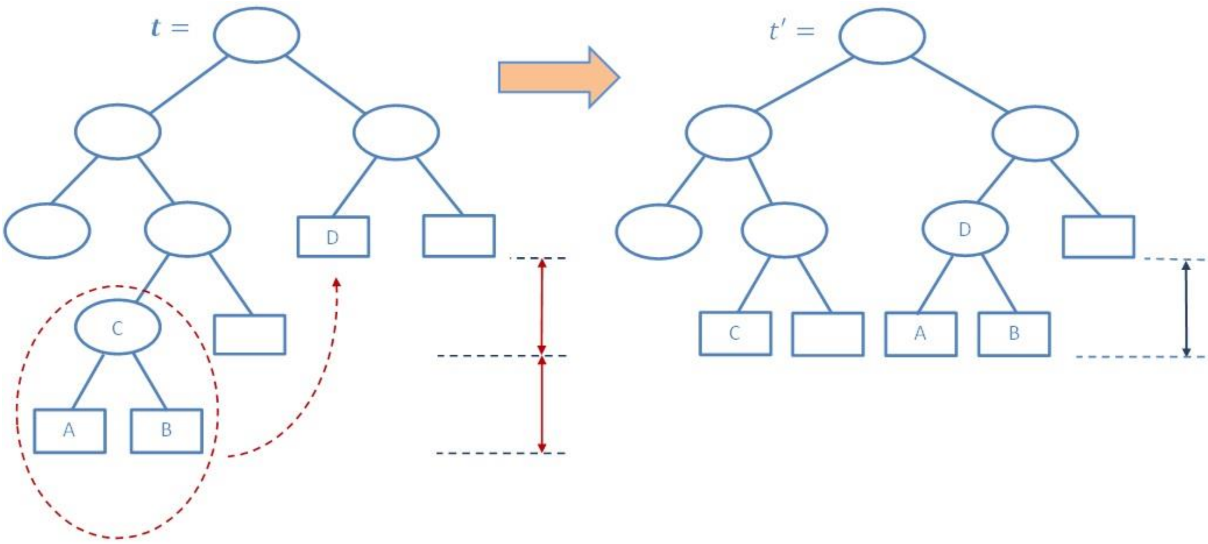
$$h(A) - h(D) = h(B) - h(D) \geq 2$$

Ha áttérünk egy olyan t' tökéletes fára, amelyet úgy kapunk, hogy az A és B leveleket, a C szülőcsúccsal együtt, legalább két szinttel magasabbra áthelyezzük, akkor a levélmagasság összeg legalább 1-gyel csökken. Az általános eset az ábra alapján könnyen meggondolható. Az ábrán látható áthelyezéssel keletkező t' fában valóban csökken az $lhsum(t)$ érték:

$$lhsum(t) - lhsum(t') = 2(h(C) + 1)) + h(D) - (h(C) + 2(h(D) + 1)) =$$
$$= 2h(C) + 2 + h(D) - h(C) - 2(h(D)) - 2 = h(C) - h(D) \geq 1$$

Világos, hogy az alsó szintű testvér levélpárok – szülővel együtt történő – *véges sokszori* áthelyezésével *majdnem teljes* tökéletes bináris fához jutunk, amelyre az $lhsum(t)$ értéke kisebb, mint bármely (ugyanannyi levélcsúcsot számláló) nem majdnem teljes tökéletes fára.

Könnyen meggondolható az is, hogy az $lhsum(t)$ érték a majdnem teljes fákra *egyértelmű*. Ha például az olyan majdnem teljes tökéletes fákat tekintjük, amelyek 6 levelet tartalmaznak, akkor egyértelmű a szintekre bontás, vagyis az, hogy az alsó szintű levelek száma 4, míg 2 levél e fölött helyezkedik el. Beszélhetünk tehát minimális vagy optimális $lhsum(t)$ értékről. ■



19.3. ábra. Egy tökéletes fa átalakítása

A lemmára támaszkodva most már kimondhatjuk a rendezések átlagos műveletigényére vonatkozó állítást.

Tétel. Bármely R összehasonlítás alapú rendező eljárás átlagosan nagyságrendben legalább $n \log n$ összehasonlítást végez az összes lehetséges bemenő sorozat rendezése során:

$$A\ddot{O}_R(n) = \Omega(n \log n)$$

Bizonyítás. Tekintsük az R összehasonlításos rendező algoritmus adott n input mérethez tartozó $t_R(n)$ *döntési fáját*. Ez a fa $n!$ számú levelet tartalmaz. Tekintsük az ugyancsak $n!$ levelet tartalmazó t_{Opt} *majdnem teljes tökéletes fát*. A lemma szerint fennáll az

$$lhsum\big(\, t_R(n)\big) \geq lhsum\big(t_{Opt}\big)$$

összefüggés. (Az optimális fa nem feltétlenül döntési fa, vagyis nem feltétlenül tartozik hozzá algoritmus, a becsléshez azonban nyilván felhasználható.) Az optimális fa magasságösszegét tovább becsüljük. A 19.4. ábra alapján világos, hogy

$$lhsum\big(t_{Opt}\big) \;\geq n!\,\big(h\big(t_{Opt}\big) - 1\big)$$

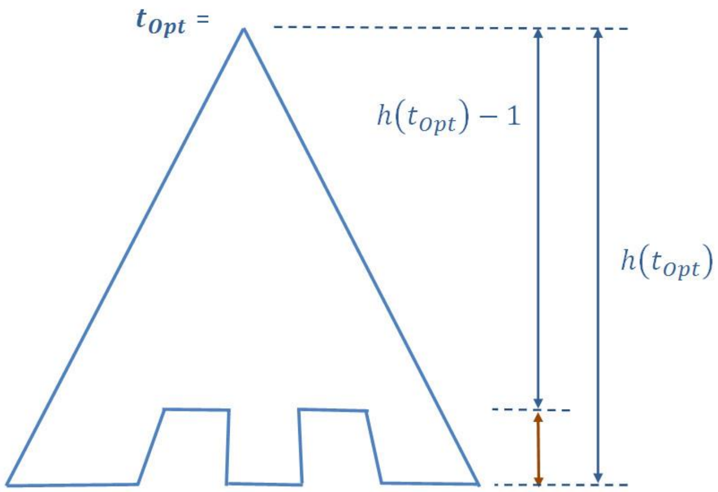
Az átlagos összehasonlítás-számra térve, annak meghatározása szerint:

$$A\ddot{O}_R(n) = \frac{lhsum\big(t_R(n)\big)}{n!}$$

Vegyük figyelembe és építsük egybe a két előző összefüggést úgy, hogy az átlagos összehasonlítás-szám definíciójából indulunk ki.

$$A\ddot{O}_R(n) = \frac{lhsum\big(t_R(n)\big)}{n!} \geq \frac{lhsum\big(t_{Opt}\big)}{n!} \geq \frac{n!\,\big(h\big(t_{Opt}\big)-1\big)}{n!} =$$
$$= h\big(t_{Opt}\big) - 1 = \Omega(n \log n)$$

Az utolsó egyenlőségnél felhasználtuk, hogy egy $n!$ számú levéllel rendelkező bináris fa magassága nagyságrendben legalább $\log_2(n!)$, ahogyan ezt az előzőekben már láttuk, hiszen a 19.2-ben bizonyított tétel állítása ezt fejezi ki. ■



19.4. ábra. A majdnem-teljes optimális tökéletes fa alakja

Ezzel a rendező eljárások összehasonlításai számára alsó korlátot adtunk mind a legrosszabb, mind az átlagos esetben. Ez az alsó korlát az $n \log n$ nagyságrend. Eredményeinket néha úgy is szokták interpretálni, hogy nem létezik *lineáris* idejű összehasonlító rendezés, ne is fáradozzunk a keresésén.