

Online ládapakolás

1. Ládapakolási modellek

A ládapakolási problémában inputként tárgyak egy sorozatát kapjuk meg, ahol az i -edik tárgyat a mérete határozza meg, ami egy $a_i \in (0, 1]$ érték. Célunk a tárgyak elhelyezése a lehető legkevesebb egység méretű ládába. Formálisabban megfogalmazva a tárgyakat olyan csoportokba akarjuk osztani, hogy minden csoportra a benne levő tárgyakra a $\sum a_i \leq 1$ feltétel teljesüljön.

Ebben a részben az online ládapakolási problémát vizsgáljuk, amely problémában az algoritmusnak az i -edik tárgyat az a_1, \dots, a_i értékek alapján kell elhelyezni a további tárgyakra vonatkozó információk ismerete nélkül. Az online algoritmusok hatékonyságát elméleti szempontból két módszerrel szokás vizsgálni. Az átlagos eset analízis során feltételezünk valamilyen típusú eloszlást a tárgyak méretéről és a használt ládák várható számát vizsgáljuk. A másik megközelítés a versenyképességi analízis, amely egy legrosszabb esetkorlát. Ebben a jegyzetben ez utóbbi módszert használjuk, ezért az alábbiakban megadjuk a pontos definíciót.

A ládapakolási problémák elemzésére a versenyképességi hányados helyett az aszimptotikus versenyképességi hányadost vizsgálják. (Az ütemezés esetén használt fogalmat a legtöbb ládapakolási modellben nem vizsgálják, ha igen akkor abszolút versenyképességi hányadosnak hívják.) Egy A algoritmus aszimptotikus versenyképességi hányadosa (R_A^∞) a következő formulákkal definiálható:

$$R_A^n = \max\{A(L)/OPT(L) \mid OPT(L) = n\}$$

$$R_A^\infty = \limsup_{n \rightarrow \infty} R_A^n.$$

Az aszimptotikus hányados fő tulajdonsága az, hogy azt vizsgálja, miként viselkedik az algoritmus akkor, ha az input mérete nagy, azaz ha az optimális költség végtelenhez tart. Ez azt jelenti, hogy az algoritmus szabadon helyezheti el a kezdeti elemeket a listáról.

A fentiekben definiált ládapakolási problémának számos változata, általánosítása van. Az alábbiakban megemlítünk néhányat. A ládapakolási probléma három különböző módon általánosítható több dimenzióra. Az első általánosítás a vektorpakolás, amelyben a tárgyak d -dimenziós vektorok és úgy kell elpakolni őket, hogy minden ládában minden koordinátára az ott szereplő értékek összege ne legyen nagyobb 1-nél. A második általánosítás a dobozpakolás, ahol többdimenziós dobozokat kell pakolni többdimenziós egységládákba. Végül a harmadik általánosítás a sávpakolás, ahol egy egységnyi széles sávba pakolunk tárgyakat és célunk, minimalizálni a használt magasságot. Fontos még megemlítenünk a ládafedési problémát, amelyben célunk, hogy a tárgyakkal a

lehető legtöbb ládát töltsük legalább egységnyi méretűre. Ezeket a modelleket itt nem tárgyaljuk, részletek találhatóak a [1] áttekintő dolgozatban.

2. Az NF algoritmus, helykorlátos algoritmusok

Érdemes még megemlítenünk azt a modellt, amelyben az egyidőben nyitott ládák száma korlátozott. Ez azt jelenti, hogy amennyiben a nyitott ládák száma eléri egy k -korlátot, akkor ezt követően csak akkor nyithatunk új ládát, ha az eddig nyitott ládák valamelyikét bezárjuk, ami azt jelenti, hogy többet nem használhatjuk. Ha a nyitott ládák száma csak egy lehet, akkor az egyetlen algoritmus, amely használható a következő NF algoritmus:

NF algoritmus: Amennyiben a tárgy elfér a nyitott ládában tegyük oda! Ellenkező esetben zárjuk be a nyitott ládát, nyissunk egy új ládát és tegyük abba a tárgyat!

Tétel Az NF algoritmus aszimptotikus versenyképességi hányadosa 2.

Bizonyítás Vegyünk egy tetszőleges σ tárgysorozatot. Jelölje n az NF algoritmus által használt ládák számát, továbbá legyen S_i , $i = 1, \dots, n$ az i -edik ládában levő tárgyak méreteinek összege. Ekkor $S_i + S_{i+1} \geq 1$, hiszen ellenkező esetben az $i + 1$ -edik láda első eleme elfért volna az i -edik ládában, ami ellentmond az algoritmus definíciójának. Következésképp a tárgyak méreteinek összege legalább $\lfloor n/2 \rfloor$. Másrészt az optimális offline algoritmus sem rakhat 1-nél több összméretű tárgyakat egy ládába, így azt kapjuk, hogy $OPT(\sigma) \geq \lfloor n/2 \rfloor$. Ez azt jelenti, hogy

$$\frac{NF(\sigma)}{OPT(\sigma)} \leq \frac{n}{\lfloor n/2 \rfloor} \leq 2 + 1/\lfloor n/2 \rfloor.$$

Másrészt ha n tart végtelenbe, akkor $1/\lfloor n/2 \rfloor$ tart a 0-hoz, amivel igazoltuk, hogy az algoritmus aszimptotikusan 2-versenyképes.

Most megmutatjuk, hogy az algoritmus aszimptotikus versenyképességi hányadosa legalább 2. Ehhez tekintsük minden n -re a következő σ_n sorozatot. A sorozat $4n$ tárgyból áll, a $2i - 1$ -edik tárgy mérete $1/2$, a $2i$ -edik tárgy mérete $1/2n$, ahol $i = 1, \dots, 2n$. Ekkor az NF algoritmus az i -edik ládába a $2i - 1$ -edik és a $2i$ -edik tárgyat teszi, és $NF(\sigma_n) = 2n$. Az optimális algoritmus az $1/2$ méretű tárgyakat párosítja, és a kis tárgyakat egy további ládába teszi, így $OPT(\sigma_n) = n + 1$. Mivel $NF(\sigma_n)/OPT(\sigma_n) = 2 - 2/(n + 1)$ a 2 értékhez konvergál, ha n tart végtelenbe, ezért igazoltuk, hogy az algoritmus aszimptotikus versenyképességi hányadosa legalább 2. \square

Itt érdemes megemlítenünk, hogy amennyiben egynél több (de korlátozott számú) láda lehet nyitva, az NF algoritmusnál jobb algoritmusok is ismertek. A jelenlegi legjobb algoritmusok a harmonikus algoritmusok családjába tartoznak, ahol az alapötlet az, hogy a $(0, 1]$ intervallumot részintervallumokra osztjuk, és minden tárgynak az az intervallum lesz a típusa, amely intervallumba a mérete esik. A különböző típusú tárgyakat különböző ládába pakoljuk, az algoritmus

párhuzamosan alkalmaz egy-egy NF algoritmust az egyes típusokhoz tartozó tárgyakra.

3. Alsó korlátok on-line algoritmusokra

Ebben a részben azt vizsgáljuk, miként találhatunk általános alsó korlátokat a lehetséges versenyképességi hányadosokra. Elsőként egy egyszerű alsó korlátot tekintünk, ezt követően megmutatjuk miként általánosítható a bizonyítás alap gondolata egy általános módszerre.

Tétel *Nincs olyan online algoritmus a ládapakolási problémára, amelynek az aszimptotikus versenyképességi hányadosa kisebb, mint $4/3$.*

Bizonyítás: Legyen A egy tetszőleges online algoritmus. Tekintsük tárgyakkal a következő sorozatát. Legyen $\varepsilon < 1/12$ és L_1 egy n darab $1/3 + \varepsilon$ méretű tárgyból álló sorozat, L_2 pedig n darab $1/2 + \varepsilon$ méretű tárgyból álló sorozat. Elsőként az algoritmus megkapja az L_1 listát. Ekkor az algoritmus bizonyos ládába két tárgyat tesz, bizonyos ládába egyet. Jelölje k azon ládák számát, amelyek két tárgyat tartalmaznak. Ekkor az algoritmus költsége $A(L_1) = k + n - 2k = n - k$. Másrészt az optimális offline algoritmus minden ládába két tárgyat tesz, így a költség $OPT(L_1) = n/2$. Amennyiben ugyanez az algoritmus az L_1L_2 összetett listát kapja, akkor az első részben szintén k ládát használ két tárgynak. (Az online algoritmus nem tudja, hogy az L_1 vagy az L_1L_2 lista alapján kapja a tárgyakat.) Következésképp az $1/2 + \varepsilon$ méretű tárgyak közül csak $n - 2k$ darabot párosíthat az előző tárgyakhoz a többihez mindhez új ládát kell nyitnia. Tehát $A(L_1L_2) \geq n - k + (n - (n - 2k)) = n + k$. Másrészt az optimális offline algoritmus minden ládába egy kisebb, $1/3 + \varepsilon$ méretű és egy nagyobb, $1/2 + \varepsilon$ méretű tárgyat tesz, így $OPT(L_1L_2) = n$. Következésképpen azt kapjuk, hogy az A online algoritmusra van olyan L lista, amelyre

$$A(L)/OPT(L) \geq \max\left\{\frac{n-k}{n/2}, \frac{n+k}{n}\right\} \geq 4/3.$$

Másrészt a fenti hányadosokban az $OPT(L)$ érték legalább $n/2$, ami tetszőlegesen nagyválasztható. Így a fenti egyenlőtlenségből adódik, hogy az A algoritmus aszimptotikus versenyképességi hányadosa legalább $4/3$, amivel a tétel állítását igazoltuk. \square

A pakolási minták módszere

A fenti bizonyítás alapötlete, hogy egy hosszabb tárgysorozatot (a fentiekben L_1L_2) veszünk és az algoritmus viselkedésétől függően választjuk ki azt a kezdőszületét a sorozatnak, amelyre a költségek hányadosa maximális. Természetes gondolat a bizonyításban használt sorozatnál bonyolultabb sorozatot használni. Több alsó korlát született különböző sorozatok felhasználásával. Másrészt a sorozatok elemzéséhez szükséges számítások egyre bonyolultabbak lettek. Az

alábbiakban megmutatjuk miként írható fel a sorozat elemzése vegyes egészértékű programozási feladatként, amely lehetővé teszi, hogy az alsó korlátot számítógép segítségével határozzuk meg.

Tekintsük a következő tárgysorozatot. Legyen $L = L_1 L_2 \dots L_k$, ahol L_i $n_i = \alpha_i n$ egyforma méretű tárgyat tartalmaz, amelyek mérete a_i . Amennyiben egy A algoritmus C -versenyképesség, akkor minden j -re teljesülnie kell a

$$C \geq \limsup_{n \rightarrow \infty} \frac{A(L_1 \dots L_j)}{OPT(L_1 \dots L_j)}$$

feltételnek. A fentiekben tekinthetjük azt az algoritmust, amelyre az általunk adható alsó korlát minimális, így célunk az

$$R = \min_A \max_{j=1, \dots, k} \limsup_{n \rightarrow \infty} \frac{A(L_1 \dots L_j)}{OPT(L_1 \dots L_j)}$$

érték meghatározása, amely érték egy alsó korlát lesz a versenyképességi hányadosra. Ezen érték meghatározható egy vegyes egészértékű programozási feladat optimumaként. A feladat megadásához szükségünk van a következő fogalmakra.

Egy tetszőleges ládára a láda tartalma leírható a láda pakolási mintájával, amely azt adja meg, hogy az egyes részlistákból hány elemet tartalmaz a láda. A *pakolási minta* egy k -dimenziós vektor (p_1, \dots, p_k) , amelynek a p_i koordinátája azt adja meg, hány elemet tartalmaz a láda az L_j részlistából. Pakolási minta olyan nemnegatív egész koordinátájú vektor lehet, amelyre a $\sum_{j=1}^k a_j p_j \leq 1$ feltétel teljesül. (Ez a feltétel azt írja le, hogy a minta által leírt tárgyak valóban elférnek egy ládában.) Osztályozzuk a lehetséges minták T halmazát a következőképpen. Minden j -re legyen T_j azon minták halmaza, amelyeknek az első pozitív együtthatója a j -edik. (Egy p minta a T_j halmazba kerül, ha $p_i = 0$ minden $i < j$ esetén, és $p_j > 0$.)

Most tekintsük az A algoritmus által kapott pakolást. Az algoritmus minden ládát valamely pakolási minta alapján töltött meg, így az algoritmus által kapott pakolás leírható a pakolási minták segítségével. Jelölje $n(p)$ minden $p \in T$ esetén azon ládák számát, amely ládákat a p mintának megfelelően pakolt az algoritmus.

Vegyük észre, hogy egy láda, amely egy a T_j osztályba eső mintának megfelelően lett megtöltve az első elemét az L_j részlistából kapja. Következésképpen azt kapjuk, hogy az algoritmus által az $L_1 \dots L_j$ részlista pakolása során kinyitott ládák száma a következőképpen adható meg az $n(p)$ értékekkel:

$$A(L_1 \dots L_j) = \sum_{i=1}^j \sum_{p \in T_i} n(p).$$

Tehát egy adott n -re a keresett A értéket a következő vegyes egészértékű programozási feladat megoldásával számíthatjuk ki.

Min R

$$\begin{aligned} \sum_{p \in T} p_j n(p) &= n_j, & 1 \leq j \leq k \\ \sum_{i=1}^j \sum_{p \in T_i} n_p &\leq R \cdot OPT(L_1 \dots L_j), & 1 \leq j \leq k \\ n(p) &\in \{0, 1, \dots\}, & p \in T \end{aligned}$$

Az első k feltétel azt írja le, hogy az összes tárgyat el kell helyeznünk a ládákbán. A második k feltétel az írja le, hogy az R érték valóban nem kisebb, mint az algoritmus költségének és az optimális költségnek a hányadosa a vizsgált részlistákra. Az $L_1 L_2 \dots L_k$ lista alapján a pakolási minták T halmaza és az optimális $OPT(L_1 \dots L_j)$ értékek meghatározhatók.

A problémában a változók igen nagy értékeket vehetnek fel és a változók száma is nagy lehet, ezért a probléma helyett a lineáris programozási relaxációt szokás tekinteni. Továbbá a megoldást azon feltétel mellett kell meghatározni, hogy n tart a végtelenbe, és ezen feltétel mellett az egészértékű feladat és a relaxáció ugyanazokat a korlátokat adják.

Az eljárást megfelelően választott listákra alkalmazva kapták meg a jelenleg ismert legjobb alsó korlátot, amely azt mondja ki, hogy nincs olyan online algoritmus, amelynek kisebb a versenyképességi hányadosa, mint 1.5401. Ezen tétel részletes bizonyítása, a módszer részletesebb tárgyalása egyéb ládapakolási alkalmazásokkal megtalálható a [4] doktori disszertációban.

4. Az FF algoritmus, a súlyfüggvény technika

Ebben a részben egy módszert mutatunk be, amely gyakran használt a ládapakolási algoritmusok analízise során. A módszert az FF (First Fit) algoritmuson mutatjuk be. Az FF algoritmus az NF algoritmus továbbfejlesztett változata, arra az esetre, amelyben nincs korlátozva a nyitott ládák száma.

FF Algoritmus: A tárgyat a legkorábban kinyitott ládába tesszük ahol elfér. Ha nem fér el egyik ládában sem, kinyitunk egy új ládát és abba rakjuk.

Itt definiáljuk a hasonló elven működő BF Best Fit algoritmust.

BF Algoritmus: A tárgyat a legnagyobb összöltéssel rendelkező ládába tesszük ahol elfér. Ha nem fér el egyik ládában sem, kinyitunk egy új ládát és abba rakjuk.

A FF algoritmusra teljesül a következő állítás.

Tétel Az FF algoritmus aszimptotikus versenyképességi hányadosa 1.7.

Bizonyítás: Mivel ezen rész fő célja a súlyfüggvény technika bemutatása, ezért pusztán azzal a résszel foglalkozunk, amely azt igazolja, hogy az algoritmus aszimptotikusan 1.7-versenyképes. A korlát élességének bizonyítása megtalálható a [3] dolgozatban. A bizonyítás alapötlete a súlyfüggvény technika, amely azt jelenti, hogy minden tárgyhoz egy súlyt rendelünk, amely azt adja meg valamilyen értelemben, hogy mennyire sok helyet foglalhat el a tárgy egy pakolásban. A tárgyaknak vesszük az összsúlyát, és ezen érték segítségével becsüljük mind az offline és az online célfüggvények értékét. Definiáljuk a következő súlyfüggvényt:

$$w(x) = \begin{cases} 6x/5, & \text{ha } 0 \leq x \leq 1/6 \\ 9x/5 - 1/10, & \text{ha } 1/6 \leq x \leq 1/3 \\ 6x/5 + 1/10, & \text{ha } 1/3 \leq x \leq 1/2 \\ 6x/5 + 2/5, & \text{ha } 1/2 < x. \end{cases}$$

A tárgyak egy tetszőleges H halmazára legyen $w(H) = \sum_{i \in H} w(a_i)$. Ekkor a súlyfüggvényre teljesülnek az alábbi állítások.

Lemma Amennyiben tárgyak egy H halmazára teljesül, hogy $\sum_{i \in H} a_i \leq 1$, akkor ezen tárgyakra $w(H) \leq 17/10$.

Lemma Tárgyak tetszőleges L listájára $w(L) > FF(L) - 2$.

Bizonyítás: Mindkét lemma bizonyítása azon alapul, hogy eseteket különböztetünk meg a tárgyak méretétől függően. A bizonyítások hosszúak és sok technikai részletet tartalmaznak, ezért jelen jegyzetben az bemutatásuktól eltekintünk. Az érdeklődő olvasó megtalálhatja a részleteket a [3] cikkben. \square

A lemmák alapján könnyen igazolható, hogy az algoritmus aszimptotikusan 1.7 versenyképes. Tekintsük tárgyaknak egy tetszőleges L listáját. Mivel az optimális offline algoritmus el tudja pakolni a lista elemeit $OPT(L)$ ládába úgy, hogy minden ládába a tárgyak méreteinek összege legfeljebb 1, ezért az első lemma alapján $w(L) \leq \frac{17}{10}OPT(L)$. Másrészt a második lemma alapján $FF(L) - 2 \leq w(L)$, így azt kapjuk, hogy $FF(L) \leq \frac{17}{10}OPT(L) + 2$, amiből következik, hogy az algoritmus 1.7-versenyképes. \square

Érdeemes megjegyeznünk, hogy számos az FF algoritmusnál jobb algoritmus került kifejlesztésre. A jelenleg ismert legjobb algoritmus aszimptotikus versenyképességi hányadosa 1.5888.

5. Többdimenziós változatok

5.1. On-line sávpakolás

A *sávpakolási feladatban* téglalapok egy halmaza adott a szélességükkel és magasságukkal, és a célunk az, hogy ezeket a téglalapokat elhelyezzük forgatások nélkül egy függőleges w szélességű sávba úgy, hogy minimalizáljuk a felhasznált rész magasságát. A továbbiakban feltételezzük, hogy a tárgyak magassága legfeljebb 1. Általában az ütemezés megosztott erőforrásokkal két dimenziót eredményez, az erőforrást és az időt. Ebben az esetben tekinthetjük a szélességet a felhasznált erőforrás nagyságának, a magasságot pedig a felhasznált időnek, így célunk a felhasznált idő minimalizálása. A feladat on-line változatát vizsgáljuk, ahol a téglalapok egy listáról érkeznek, és a megérkezett téglalapot el kell helyezni a függőleges sávban a további téglalapokra vonatkozó ismeretek nélkül. Az on-line sávpakolási feladatra kidolgozott algoritmusok többsége a polc algoritmusok családjába tartozik. az alábbiakban ezt az algoritmuscsaládot ismertetjük.

POLC algoritmusok

Egy alapvető módszer a téglalapok pakolására az, hogy polcokat definiálunk és a téglalapokat ezekre a polcokra helyezük el. *Polcon* a feltöltendő sávnak egy vízszintes részét értjük. A POLC algoritmus minden téglalapot egy polcra helyez. Miután az algoritmus kiválasztotta azt a polcot, amely a téglalapot tartalmazni fogja, az algoritmus a téglalapot elhelyezi a polcon annyira balra, amennyire lehetséges a már a polcon levő egyéb téglalapok átfedése nélkül. Tehát a téglalap érkezése után az eljárásnak két döntést kell hoznia. Az első döntés az, hogy az eljárás kialakít-e egy új polcot vagy sem. Ha új polcot alakítunk ki, meg kell határoznunk a polc magasságát is. Az újonnan kialakított polcokat mindig az előző polc tetejére helyezük, az első polc a sáv legalján van. A második döntés, hogy az algoritmusnak ki kell választani azt a polcot, amelyre a téglalapot helyezi. A továbbiakban akkor mondjuk, hogy egy *téglalap elhelyezhető* egy polcon, ha a polc magassága nem kisebb a téglalap magasságánál és a polcon elég hely van ahhoz, hogy a téglalapot elhelyezzük rajta.

Egy eljárást vizsgálunk részletesen a fenti feladat megoldására. Ez az algoritmus a Baker és Schwarz által 1983-ban kifejlesztett NFS_r algoritmus. Az algoritmus egy $r < 1$ paramétertől függ. Az algoritmus minden j -re legfeljebb egy r^j magasságú aktív polcot tart fent és egy tárgy érkezése után a következő szabállyal definiálhatjuk.

A $p_i = (w_i, h_i)$ téglalap érkezése után válasszunk egy olyan k értéket, amelyre teljesül, hogy $r^{k+1} < h_i \leq r^k$. Amennyiben van r^k magasságú aktív polc, és a téglalap elhelyezhető ezen a polcon, akkor helyezük el. Ellenkező esetben alakítsunk ki egy új r^k magasságú polcot, helyezük el a téglalapot rajta, és a továbbiakban legyen ez a polc az r^k magasságú aktív polc (ha volt korábbi aktív polc, azt lezárjuk).

példa Legyen $r = 1/2$. Legyen az első tárgy mérete $(w/2, 3/4)$. Ez a tárgy 1 magasságú polcra kerül. Ekkor létrehozunk egy 1 magasságú polcot a sáv legalján, ez lesz az 1-magasságú aktív polc, és ennek a polcnak a bal sarkára helyezük el a tárgyat. Legyen a következő tárgy mérete $(3w/4, 1/4)$. Ez a tárgy $1/4$ magasságú polcra kerül. Mivel nincs ilyen aktív polc, ezért létrehozunk egy $1/4$ magasságú polcot az előző 1 magasságú polc tetején, ez lesz az $1/4$ magasságú aktív polc, és ennek a polcnak a bal sarkára helyezük el a tárgyat. Legyen a következő tárgy mérete $(3w/4, 5/8)$. Ez a tárgy ismét 1 magasságú polcra kerül. Mivel az 1 magasságú aktív polcon nem fér el, ezért azt lezárjuk, és létrehozunk egy új 1 magasságú polcot az előző $1/4$ magasságú polc tetején, ez lesz az 1 magasságú aktív polc, és ennek a polcnak a bal sarkára helyezük el a tárgyat. Legyen a következő tárgy mérete $(w/8, 3/16)$. Ez a tárgy $1/4$ magasságú polcra kerül. Az $1/4$ magasságú aktív polcon még elfér a tárgy, ezért arra polcra rakjuk, annyira balra, amennyire lehetséges a második tárgy mellé.

NFS_r versenyképességére igazak az alábbi állítások.

Tétel Az NFS_r algoritmus $(\frac{2}{r} + \frac{1}{r(1-r)})$ -versenyképes. Az NFS_r algoritmus aszimptotikusan $2/r$ -versenyképes.

Bizonyítás Tekintsük téglalapoknak egy tetszőleges L listáját, jelölje H a legmagasabb téglalap magasságát. Mivel ekkor $\text{OPT}(L) \geq H$ teljesül, ezért a tétel állításának igazolásához elegendő belátnunk, hogy erre a sorozatra

$$\text{NFS}_r(L) \leq 2/r \text{OPT}(L) + H/r(1-r).$$

Jelölje H_A az L lista végén az aktív polcok összmagasságát, H_Z pedig a többi lezárt polcok összmagasságát. Elsőként vizsgáljuk az aktív polcokat. Jelölje h a legmagasabb aktív polc magasságát. Ekkor az aktív polcok magasságai a hr^i értékeket vehetik fel és minden i -re legfeljebb egy aktív polc van hr^i magassággal. Tehát az aktív polcok összmagasságára

$$H_A \leq h \sum_{i=0}^{\infty} r^i = \frac{h}{1-r}.$$

Másrészt a $H > rh$ egyenlőtlenségnek is teljesülnie kell, hiszen különben a legmagasabb téglalap is elért volna a legfeljebb rh magasságú polcokon és nem nyitottuk volna meg a h magasságú polcot. Következésképpen $H_A \leq H/r(r-1)$.

Most vizsgáljuk a lezárt polcokat. Vegyük egy tetszőleges i -re a hr^i magasságú polcokat, jelölje ezeknek a számát n_i . Minden ilyen lezárt S polcra a következő S' polc elsőként egy olyan elemet tartalmaz, amely már nem volt elhelyezhető, így a két egymást követő polcra az elhelyezett téglalapok teljes szélessége legalább w . Másrészt a hr^i magasságú polcokon minden tárgy magassága legalább hr^{i+1} , hiszen egyébként a tárgyat egy kisebb magasságú polcra helyeznénk. Tehát párosítva a lezárt polcokat és használva az aktív hr^i magasságú polcot is, ha a lezárt polcok száma páratlan, azt kapjuk, hogy az ilyen polcokon elhelyezett tárgyak összterülete legalább $wn_ihr^{i+1}/2$. Következésképpen az összes téglalapnak az összterülete legalább $\sum_{i=0}^{\infty} wn_ihr^{i+1}/2$, így $\text{OPT}(L) \geq \sum_{i=0}^{\infty} n_ihr^{i+1}/2$. Másrészt a lezárt polcok összmagassága $H_Z = \sum_{i=0}^{\infty} n_ihr^i$, így azt kaptuk, hogy $H_Z \leq 2\text{OPT}(L)/r$. Mivel $\text{NFS}_r(L) = H_A + H_Z$, ezért a fentiek alapján adódik a kívánt egyenlőtlenség. \square

A fenti algoritmuson kívül további polc algoritmusokat is vizsgáltak a feladat megoldására. A fenti algoritmus alap gondolata, hogy az egyes polctípusokat ládaként fogjuk fel, és az adott polctípushoz rendelt tárgyakat az NF ládapakolási algoritmusmal helyezzük el. Természetes gondolat más ládapakolási algoritmusok használata. A jelenlegi legjobb polc algoritmust Csirik és Woeginger fejlesztették ki 1997-ben, amely algoritmus a harmonikus ládapakolási algoritmust használja az adott polctípusokhoz rendelt tárgyak elhelyezésére.

5.2. On-line sávpakolás nyújtható tárgyakkal

Amennyiben a sávpakolási feladattal azt az erőforrás allokációs problémát modellezük, amelyben a tárgyak szélessége az erőforrásigény a tárgyak magassága az idő, akkor használhatjuk azt a változatot, amelyben a tárgyak mérete nem fix, hanem az egyes tárgyakat megnyújthatjuk a területüket fixen hagyva (ld [2]). Ebben a részben az egyszerűbb tárgyalás érdekében feltesszük, hogy a

sáv szélessége 1. Egyszerűen látható, hogy ebben az esetben az offline probléma könnyen megoldható. Az offline optimum értéke $OPT(L) = \max\{S, H\}$, ahol $S = \sum_{i \in L} w_i \cdot h_i$ a teljes terület, és $H = \max_{i \in L} h_i$ a maximális magasság. Az aszimptotikus versenyképességi hányadost használva tetszőlegesen közel juthatunk az 1 értékhez ha nagyon nagy additív konstans engedünk meg az algoritmus elemzésében, következésképpen ebben az esetben a versenyképességi hányadost érdemes használni. A feladat megoldására egyből adódik az NFS_r algoritmus következő módosítása.

A $p_i = (w_i, h_i)$ téglalap érkezése után válasszunk egy olyan k értéket, amelyre teljesül, hogy $r^{k+1} < h_i \leq r^k$. Nyújtsuk meg a téglalapot r^k magasságúra. Amennyiben van r^k magasságú aktív polc, és az új téglalap elhelyezhető ezen a polcon, akkor helyezzük el. Ellenkező esetben alakítsunk ki egy új r^k magasságú polcot, helyezzük el a téglalapot rajta, és a továbbiakban legyen ez a polc az r^k magasságú aktív polc (ha volt korábbi aktív polc, azt lezárjuk).

Az így kapott algoritmusra igaz a következő állítás.

Tétel Az NFS_r algoritmus $(2 + \frac{1}{r(1-r)})$ -versenyképes.

Bizonyítás Tekintsük téglalapoknak egy tetszőleges L listáját, jelölje H a legmagasabb téglalap magasságát. Mivel ekkor $OPT(L) \geq H$ teljesül, ezért a tétel állításának igazolásához elegendő belátnunk, hogy erre a sorozatra

$$NFS_r(L) \leq 2OPT(L) + H/r(1-r).$$

Jelölje H_A az L lista végén az aktív polcok összmagasságát, H_Z pedig a többi lezárt polcok összmagasságát. Elsőként vizsgáljuk az aktív polcokat. Jelölje h a legmagasabb aktív polc magasságát. Ekkor az aktív polcok magasságai a hr^i értékeket vehetik fel és minden i -re legfeljebb egy aktív polc van hr^i magassággal. Tehát az aktív polcok összmagasságára

$$H_A \leq h \sum_{i=0}^{\infty} r^i = \frac{h}{1-r}.$$

Másrészt a $H > rh$ egyenlőtlenségnek is teljesülnie kell, hiszen különben a legmagasabb téglalap is elért volna a legfeljebb rh magasságú polcokon és nem nyitottuk volna meg a h magasságú polcot. Következésképpen $H_A \leq H/r(r-1)$.

Most vizsgáljuk a lezárt polcokat. Vegyük egy tetszőleges i -re a hr^i magasságú polcokat, jelölje ezeknek a számát n_i . Minden ilyen lezárt S polcra a következő S' polc elsőként egy olyan elemet tartalmaz, amely már nem volt elhelyezhető, így a két egymást követő polcra az elhelyezett téglalapok teljes szélessége legalább 1. Továbbá a téglalapokat megnyújtjuk mielőtt az aktuális polcra raknánk, így minden téglalap magassága pontosan hr^i . Tehát párosítva a lezárt polcokat és használva az aktív hr^i magasságú polcot is, ha a lezárt polcok száma páratlan, azt kapjuk, hogy az ilyen polcokon elhelyezett tárgyak összterülete legalább $n_i hr^i/2$. Következésképpen az összes téglalapnak az összterülete legalább $\sum_{i=0}^{\infty} n_i hr^i/2$, így $OPT(L) \geq \sum_{i=0}^{\infty} n_i hr^i/2$. Másrészt a lezárt polcok

összmagassága $H_Z = \sum_{i=0}^{\infty} n_i h r^i$, így azt kaptuk, hogy $H_Z \leq 2OPT(L)$. Mivel $NFS_r(L) = H_A + H_Z$, ezért a fentiek alapján adódik a kívánt egyenlőtlenség. \square

Ennél jobb versenyképességi hányadossal rendelkezik a következő DUPLAZO algoritmus.

DUPLAZO ALGORITMUS

Az első téglalap érkezése után vegyünk egy h_1 magas polcot a sáv legalján és legyen ez az aktív polc. A további elemeket pakoljuk az alábbi szabály szerint:

1. *Lépés* Ha lehetséges a téglalapot berakni az aktív polcra, azt követően, hogy megnyújtjuk az aktív polc magasságára, akkor nyújtjuk meg és rakjuk az aktív polcra annyira balra, amennyire lehetséges. Egyébként 2. lépés.

2. *Lépés* Vegyünk egy új aktív polcot, amely kétszer olyan magas, mint az előző tegyük az eddigi polcok tetejére. Lépünk az 1. lépésre.

Az algoritmusra teljesül a következő állítás.

Tétel A DUPLAZO algoritmus versenyképességi hányadosa 4.

Bizonyítás: Elsőként igazoljuk, hogy az algoritmus 4 versenyképes. Vegyünk egy tetszőleges L listáját téglalapoknak. Legyen H az utolsó aktív polc magassága. Amikor az algoritmus ezt a polcot megkonstruálta, akkor az aktuális téglalap nem fért el az előző $H/2$ magas aktív polcon, következésképpen $H \leq 2 \cdot OPT(L)$. Másrészt a használt polcmagasságok $H/2, H/4, \dots, H/2^i$ valamely i -re, így a teljes sávmagasság legfeljebb $2H$.

A korlát élességének igazolásához vegyük a következő L_i listát. Az első i téglalap legyen $(1/4, 2^j)$, $j = 1, \dots, i$, az utolsó pedig legyen $(1/4, 2^i + 1)$. Végrehajtva az algoritmust, és megkonstruálva az optimális offline megoldást azt kapjuk, hogy $DS(L_i)/OPT(L_i) = 2^{i+2}/(2^i + 1)$, ami tart 4-hez, amint i tart a végtelenbe.

Hivatkozások

- [1] J. Csirik, G. Woeginger, On-line packing and Covering problems, *Online algorithms: The State of the Art Vol. 1442 of Lecture Notes in Computer Science*, Fiat A., G. J. Woeginger (eds.), Springer-Verlag, Berlin - Heidelberg, 1998 147–177
- [2] Cs. Imreh, Online strip packing with modifiable boxes, *Operation Research Letters*, **66**, 79-86, 2001.
- [3] D. S. Johnson, A. Demers, J. D. Ullman, R. M. Garey, R. L. Graham, Worst-case performance bounds for simple one-dimensional packing algorithms, *SIAM J. of Computing* **3** 256–278, 1974.
- [4] A. van Vliet, *Lower and upper bounds for on-line bin packing and scheduling heuristics*, PhD thesis, Erasmus University, Rotterdam, The Netherlands, 1995.