

József Attila Tudományegyetem, SZEGED

1 9 8 2

A DARABOLÁSI FELADATOK ÉS A MEGOLDÁSUKRA ALKALMAZOTT
MÓDSZEREK ÁTTEKINTÉSE

Egyetemi doktori értekezés

Lengyel Imre
matematikus

Dél-Alföldi TCSV, Békéscsaba

T a r t a l o m

1. <u>Bevezetés</u>	4.
1.1. Alapfogalmak	5.
1.2. A darabolás lineáris programozási feladata	8.
1.3. Megjegyzések	11.
2. <u>A GILMORE-GOMORY módszer</u>	13.
2.1. A módszer alapgondolata.	13.
2.1.1. A darabolási feladat jellemző feltételei	21.
2.2. Az egy-dimenziós hátizsákfeladatok megoldási módszerei	27.
2.3. A két-dimenziós darabolási feladatok	36.
2.3.1. Két-ütemű quillotine-vágások	37.
2.3.2. Az általános quillotine-vágás.	42.
2.4. Javaslatok a GGM-hez	52.
2.5. A GGM alkalmazásai	55.
3. <u>Egy-dimeziós darabolási feladat nem-lineáris cél- függvény esetén</u>	60.
3.1. A nem-lineáris darabolási feladatok	61.
3.2. A szekvenciális heurisztikus eljárás.	65.
3.3. Módosítások és alkalmazások	71.



4. <u>Nagyméretű alaptáblák heurisztikus feldarabolása</u>	76.
4.1. Az al-táblák feltöltése	78.
4.2. Az alaptábla felosztása al-táblákra.	81.
5. <u>A síkűveg szabásának optimalizálása az Orosházi Üveggyárban.</u>	84.
5.1. Az üvegszabási probléma leírása.	87.
5.2. A leszabás matematikai modellezése	89.
5.2.1. Az összes szabáskép meghatározása.	90.
5.2.2. Az egészértékű lineáris programozási feladat	99.
5.3. A modell számítógépes megvalósítása.	101.
5.3.1. Eljárás az összes szabáskép előállítására.	103.
5.3.2. Az ILP feladat megoldása	108.
5.4. Futtatási eredmények	108.
6. <u>A GGM gyakorlati megvalósítása</u>	114.
6.1. A készletfeltétel kezelése	114.
6.2. A GGM-t realizáló programcsomag programterve	120.
6.3. Futtatási eredmények	124.
7. <u>Egyéb megközelítések</u>	129.
Összefoglalás	134.
Hivatkozások.	135.

Az üzemekben gyakran jelentkező darabolási /kiszabási, levágási, nyirási, stb./ feladatok megoldásának matematikai-számítástechnikai módszereivel foglalkozunk, amikor eltérő méretű idomokat kell valamilyen alapanyagból levágni, kiszabni. Ilyenkor a cél olyan szabásminták megadása, amelyek alkalmazása esetén az alapanyagok feldarabolásának a költsége minimális lesz. Ez a feladat előfordul az üvegiparban, ahol a raktáron levő nagyméretű alaptáblákat kell feldarabolni az igényelt méretű és mennyiségű rendelés szerint, a papíriparban papirtekercsek felvágásakor vagy a fémipar vállalatoknál fémlemezek hasításakor, tekercsek nyirásakor, stb.

A feladat tulajdonképpen valamely alakzat ugyanolyan dimenzióju nem-átfedő alakzatokkal való lefedése. Hasonlóan modellezhető pl. a hajók /vagy ládák/ térfogatának kitöltése, mágneslemezen a file-k elhelyezése vagy adott erőforrások szétosztása a felhasználók között.

A dolgozatban az alapanyagok feldarabolásának speciális feltételeivel, az ezeken alapuló, az ezeket kielégítő egzakt és heurisztikus módszerekkel és algoritmusokkal foglalkozunk. Az irodalomban ismertetett alkalmazások és módszerek bemutatása után részletesen vázolunk két eltérő elméleti alapokon álló módszert felhasználó, a közreműködésünkkel készült gyakorlati alkalmazást és az adaptálásukat számítógépre.

1. Bevezetés

Az ötvenes évek végére igény és lehetőség mutatkozott arra, hogy számítógéppel határozzák meg a legelőnyösebb szabásmintákat. Legkézenfekvőbbnek a lineáris programozás mutatkozott, azonban a nagyméretű feladat kezelése gyakorlati nehézségekbe ütközött, másrészt gondokat okozott a darabolás diszkrét /egészértékű/ jellege is. Ezért volt úttörő jelentőségű GILMORE és GOMORY cikksorozatban [33-36] ismertetett módszere, amellyel a darabolás nagyméretű lineáris programozási feladatát egzaktul, viszonylag kis helyen és gyorsan meg lehet oldani. Módosított szimplexmódszert alkalmaztak, ahol a bázisba bevonandó vektort egy kiegészítő hátizsákfeladat megoldásaként határozták meg. Ez a megközelítés rendkívül ötletesen a darabolási feladatok egy széles osztályára egzakt, a gyakorlatban hasznosítható megoldást ad és az alkalmazások döntő többségének az alapját képezi. A hatvanas évektől széleskörűen hasznosították /illetve hasznosítják/ ezeket az eredményeket. Mivel a darabolások üzemi, műszaki feltételei rendkívül eltérők, ezért az alkalmazások elterjedésével egyidőben a hetvenes évektől speciális - főleg egyedi feltételeket figyelembe vevő heurisztikus - eljárások tömegét készítették el. Ezek a módszerek elsősorban olyan darabolási feladatokra készültek, amelyeket lineáris programozással nem, vagy csak igen durva közelítéssel oldhatók meg.

A dolgozatban az alapfogalmak, az alapvető lineáris programozási feladat ismertetése után részletesen bemutatunk három különböző megközelítést. Először a legalapvetőbb módszert ismertetjük, amely GILMORE és GOMORY eredményein, munkáin alapul. Azt követően két speciális feltételeket figyelembe vevő heurisztikus eljárást vázolunk. Az első a harmadik fejezetben bemutatott egy-dimenziós, nem-lineáris célfüggvénnyel bíró feladat HAESSLER által adott megoldás, amely főleg papir- és acéltekercsek darabolásakor alkalmazható előnyösen. A másik az a negyedik fejezetben ismertetett dinamikus programozáson alapuló, két-dimenziós darabolási feladatra alkalmazható eljárás, amely nagymértékű alapanyagok esetén használható. Ez az eljárás ADAMOWICZ, ALBANO és ORSINI munkáin alapszik. Ezt követően előbb az Orosházi Üvegyár részére készített, alaptáblák fel-darabolását optimalizáló modellt és a számítógépes próba-futtatásokat, majd a GILMORE-GOMORY módszer egy lehetséges realizálását végző programrendszert mutatunk be. A továbbiakban röviden vázoljuk az egyes speciális problémák feltételeit és a megoldásuk alapelveit. A számítógépes futtatások közül csak a leglényegesebb - összevetési alapként is figyelembe vehető - eredményeket ismertetjük.

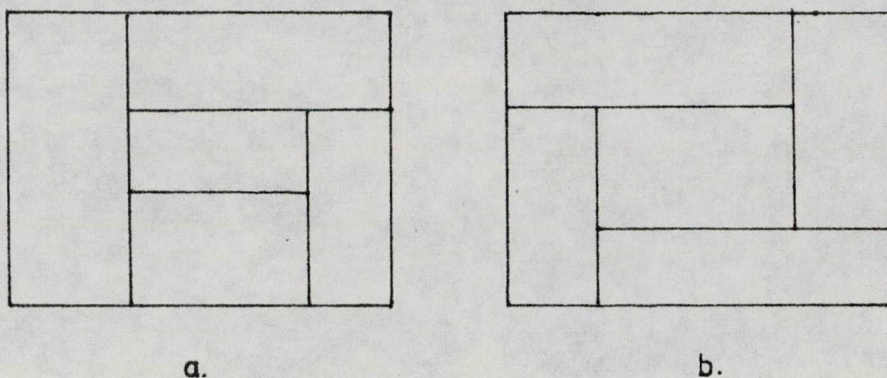
1.1. Alapfogalmak

A darabolási feladatokat csoportosíthatjuk az alapanyagok a vágásnál figyelembe vett dimenzióinak megfelelően. Egy-dimenziós a feladat például csövek, rudak, szalagok, papir- vagy

acéltekercsek feldarabolásakor, amikor az alapanyagnak csak az egyik méretét kell figyelembe venni, mivel a többi mérete megegyezik a rendelések méreteivel. Két-dimenziós a feladat, amikor például üvegtáblákat, fémlemezeket, butorlapokat, azaz valamilyen síkidomot kell kisebb síkidomokra felszabni. A térfogatkitöltési, rakodási, csomagolási stb. feladatok a három-dimenziós darabolásnak felelnek meg.

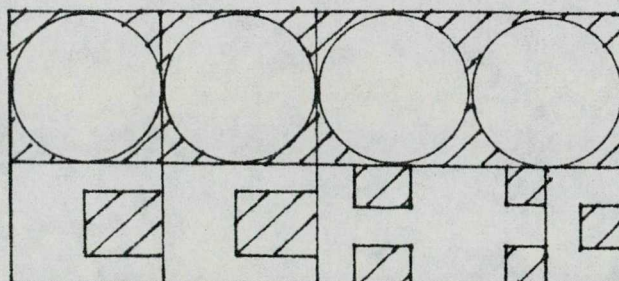
Dolgozatunkban törekszünk az egy dimenziós darabolás összes lényegi - a legtöbbet idézett - feltételének az ismertetésére. A két-dimenziós feladatoknak egy speciális, a gyakorlatban a legtöbbet előforduló esetével foglalkozunk részletesen, amikor téglalap alakú alapanyagból /alaptáblából/ kell kisebb téglalapokat /téglalapból-téglalapot/ az alapanyagnak valamelyik oldalára merőleges vágással /ezt a vágást ortogonálisnak nevezzük/ leszabni. Továbbá a vágás az alaptáblán széltől-szélig történik, azaz csikot /vagy csikokat/ vágunk először, majd ezt daraboljuk tovább. Ezt a darabolási módszert quillotine-vágásnak nevezik /l. ábra/. Ez az iteratív, illetve rekurzív eljárások használatát sugallja, mivel egy vágás után újra téglalap alakú, kisebb méretű "alapanyagot" kapunk.

Ha a kért idom nem téglalap alakú, akkor ezt az idomot lefedő legkisebb téglalapot véve és kiszabva, a feladatot visszavezetjük a téglalapok esetére. Erre a visszavezetésre HAIMS és FREEMAN [51] dinamikus programozáson alapuló eljárást javasolt,



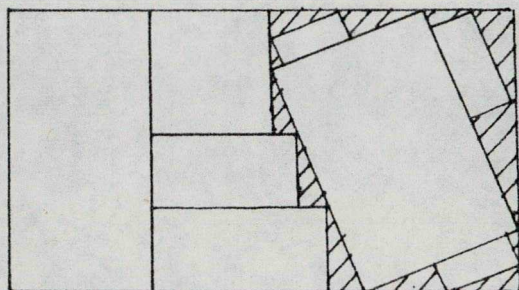
1. ábra. Példa quillotine- /a/ és nem-quillotine-vágásra /b/.

és természetesen bármelyik téglalapról-téglalapot algoritmus alkalmazható. Téglalapról szabálytalan idomok kiszabására /2. ábra/ - téglalapokra való visszavezetés nélkül - a 4. fejezetben röviden utalunk, de nem részletezzük, mivel ismertetésük meghaladja a dolgozat kereteit. Megjegyezzük, hogy a szabálytalan alakok kivágására egyre biztatóbb eredmények születnek [9], [96], [97], sőt programozott szabásgépeket is árusítanak [75].



2. ábra. Szabálytalan idomok leszabása téglalapok vágására visszavezetve.

A téglalapból-téglalapot vágás olyan speciális eseteire is vannak már gyakorlati próbálkozások [22] és elméleti megközelítések [26], amikor az alaptáblának az oldalára nem merőlegesek a vágások /3. ábra/, azaz nem ortogonális a kiszabás.



3. ábra. Példa két-dimenziós nem-ortogonális vágásra

A három- és több-dimentizós feladatra nem térünk ki.

1.2. A darabolás lineáris programozási feladata

A darabolási feladattal elsőként KANTOROVICS foglalkozott híres munkájában [70], ahol egészértékű lineáris programozási feladat /ILP/ optimális megoldásaként kapta meg az optimális változatokat. A következőkben az ILP-t ismertetjük, amelynek a felépítése egy- és két-dimenziós esetben is ugyanaz, ezért dimenziók nélkül fogalmazzuk meg.

Tegyük fel, hogy adott azoknak az alapanyagoknak a mérete, mennyisége és költsége, amelyek feldarabolásával kell az ismert méretű és tétel számu /igényelt darabszámú/ rendeléseket megkapnunk. Szabásmintán /szabásképen, szabásrajzon, leszabási változaton, stb./ a rendelések egy részének nem-átfedő elrendezését értjük az alapanyagon. A darabolási feladat: olyan szabásképek sorozatának a megadása, amelyek alkalmazása esetén minden rendelést a tétel számának megfelelően kielégítünk és a felhasznált alapanyagok összköltsége minimális.

Legyen a rendelések sorrendje rögzített és a számuk: m . Tegyük fel, hogy ezekből a rendelésekből összeállítható összes lehetséges leszabási változatot ismerjük, a számuk: n . Egy szabásképen belül a rendelések sorrendje, elhelyezkedése az optimalizálás szempontjából tetszőleges, ezért a szabásképet egyértelműen meghatározza az, hogy melyik rendelésből hány szerepel benne /4a. ábra/. Egy adott szabásmintához hozzárendelünk egy m -dimenziós oszlopvektort, amelynek i -edik komponense azt mutatja, az i -edik rendelésből hány szerepel ebben a szabásképen. Ezeket az oszlopvektorokat szabásvektoroknak nevezzük /lásd 4. ábra, ahol X a transzponálás jele/.

Az optimális szabásmintákat /és azt, hogy hányszor kell alkalmazni őket/ a következő ILP optimális megoldásaként kapjuk:

$$x_j \geq 0, x_j = \text{egész} \quad /j = 1, 2, \dots, n/$$

$$M/ \sum_{j=1}^n a_{ij} x_j \geq N_i \quad /i = 1, 2, \dots, m/$$

$$\sum_{j=1}^n c_j x_j \rightarrow \min!$$

ahol n = a szabásképek száma,

m = az eltérő méretű rendelések száma,

N_i = az i -edik rendelés tételszáma,

a_{ij} = azt mutatja, hogy az i -edik rendelésből a
 j -edik szabásképből hány szerepel,

c_j = a j -edik szabásminta alapanyagának a költsége,

x_j = változó értéke azt mutatja, hogy a j -edik szabás-
képet hányszor kell alkalmazni.

Az ILP megoldásával az összes lehetséges leszabási változatból kiválasztjuk a legmegfelelőbb szabásmintákat, amelyeknek szabás-vektorai az ILP optimális megoldásának pozitív elemeihez tartozó vektorok. Az ILP optimális megoldásának pozitív elemei azt mutatják, hogy a megfelelő szabásmintát hányszor kell alkalmazni, a célfüggvény optimális értéke pedig a felhasznált alapanyagok összes költségét jelenti.

Az ILP megoldása a gyakorlatban nehézségeket okoz, mivel

/i/ a szabásképek száma nagyon nagy,

/ii/ a feladat egészértékű.

Az /i/ miatt KANTOROVICS a feladatot a gyakorlatban megoldhatatlannak tartotta. A szabásképek számának a nagyságrendjéről tájékoztat a [34] -ban található kombinatorikus becslés: egy-dimenziós esetben, ha 200 egység hosszúságu alapanyagokból 40 különböző hosszúságu rendelést vágunk le, amelyek hossza legalább 20 és legfeljebb 40 egység, akkor a leszabási változatok száma 10 és 100 milliárd között van. Ilyen nagyméretű feladat megoldása valóban megvalósíthatatlannak tűnik, márpedig ilyen jellegű darabolási probléma a gyakorlatban sűrűn felvetődik.

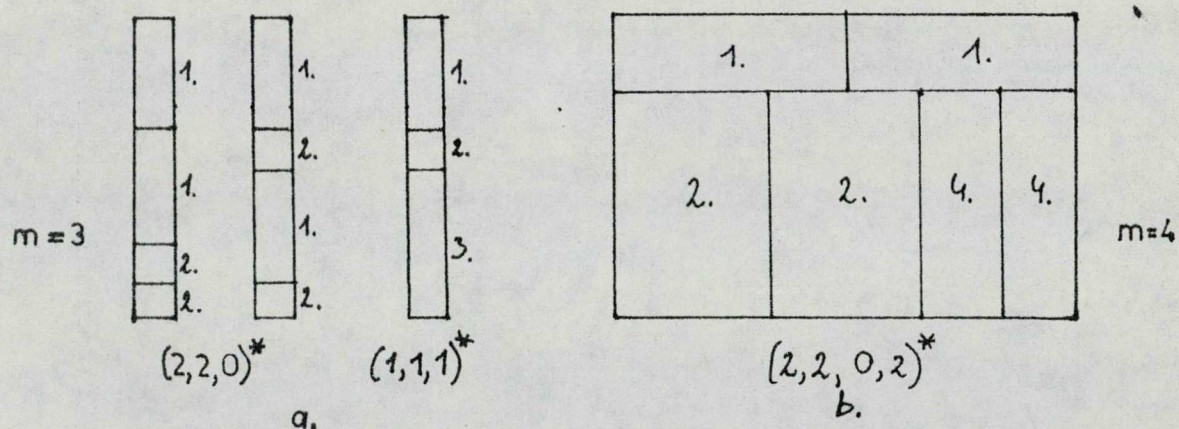
A hazai alkalmazások egy részénél [9] , [77] , [79] a "legelőnyösebb" szabásképeket elkészítik és az ILP-t közvetlenül megoldják /illetve a folytonos feladatot oldják meg és felfelé kerekítik a nem-egész megoldást. Az 5. fejezetben egy ilyen megközelítést ismertetünk/. Ez a megközelítés akkor célszerű, ha a szabásképek száma a műszaki, technológiai korlátok miatt legfeljebb annyi, hogy a feladatot egy legalább közepes méretű számítógép még kezelni tudja és a szabásképeket gyorsan elő lehet állítani /lásd [79] , lényeges szempont még az is, hogy ilyen esetben az alapanyagok raktárkészletét egyszerűen figyelembe lehet venni/. Ekkor egy ILP-t realizáló könyvtári programcsomaggal az optimális megoldás megkereshető.

1.3. Megjegyzések

Az ILP-ben természetesen a technológiai, műszaki körülményeket lineáris feltételként kell megfogalmazni. Ilyen körülmény például: a rendelkezésre álló eltérő méretű alapanyagok

készlete. Előfordulhatnak azonban olyan feltételek is, amelyek lineáris alakban nem írhatók fel, például: az optimális szabásminták száma egy adott korlát alatt legyen, amely korlát kisebb, mint m /azaz $\sum_i \text{sign } x_i \leq m$ /. Nyilván a nem-lineáris feltételek kezelése egészen más megfontolásokat, megoldási módszereket igényel.

A célfüggvényben a legtöbb esetben c_j a j -edik szabásminta alapanyagának a költségét jelenti. Amennyiben a szabásgép használata szabásmintaként eltérő költséget von maga után, akkor azt is be kell építeni az együtthatóba. Ha a c_j a százalékos hulladékot jelenti, akkor előfordulhat, hogy bizonyos rendelésekből a tételszámánál jóval többet szabunk le, azaz raktárra termelünk. A darabolások gyakorlati feltételeire és matematikai megfogalmazásaikra később az egyes módszerek bemutatása során kitérünk.



4. ábra: A szabásképen belül a rendelések elhelyezkedése a optimalizálás szempontjából tetszőleges.

2. A GILMORE-GOMORY módszer

GILMORE és GOMORY 1961-ben az egy-dimenziós darabolás ILP feladatának /1/ megoldására hatékony, a gyakorlatban jól alkalmazható eljárást adott [33]. Az alapgondolatukban egészértékűségtől eltekintettek és azt használják fel, hogy egy adott bázis esetén a bázisba bevonandó vektor az szabás-vektor, így csak a legmegfelelőbb szabás-vektort kell kiválasztani. Ez a kiválasztás pedig visszavezethető egy hátizsákfeladatra. Tehát módosított szimplexmódszert alkalmazva - ahol egy induló bázis-megoldás ismert -, hátizsákfeladatok sorozatát megoldva megkapjuk a nagyméretű, folytonos lineáris programozási feladat optimális megoldását. Eredményeiket későbbi munkáikban továbbfejlesztették és a darabolási feladatok szinte összes jellemző vetületét, matematikai háttérét megvizsgálták [34], [35]. A speciális hátizsákfeladatok megoldására pedig kidolgozták a dinamikus programozáson alapuló ún. "hátizsák-függvények" elméletét [36].

A következőkben a GILMORE-GOMORY módszert /GGM/ és alkalmazásait ismertetjük egy-, majd két-dimenziós esetben. A módszer egy lehetséges számítógépes alkalmazását a 6. fejezetben mutatjuk be.

2.1. A módszer alapgondolata

Az /1/ ILP-t úgy oldjuk meg, hogy az előző fejezetben említett két probléma közül az egészértékűségtől eltekintünk, a

folytonos LP optimális megoldásának nem-egészértékű elemeit felfelé kerekítjük és "megfelelőnek" fogadjuk el az így kapott szabásmintákat. Szuboptimális megoldáshoz jutunk, azonban a gyakorlati példák igazolják [34] , [69] , hogy ez a kerekítés nagymennyiségű alapanyag feldarabolása esetén - ami az üzemi feladatok többségénél teljesül - nem okoz számottevő eltérést az ILP "optimális" megoldásától, mivel ilyenkor egy szabásképet elég sokszor kell alkalmazni.

A szabásképek nagy számából adódó nehézségeket az alábbi módon küszöbölték ki. Irjuk át /1/ -t folytonos lineáris programozási feladattá segédváltozókat bevezetve a következő alakba:

$$x_j \geq 0 \quad /j = 1, 2, \dots, n, n+1, \dots, n+m/$$

$$/2/ \quad \sum_{j=1}^n a_{ij} x_j - x_{n+i} = N_i \quad /i = 1, 2, \dots, m/$$

$$\sum_{j=1}^n c_j x_j \longrightarrow \min!$$

Az eljárás alapgondolata - amely a módosított szimplex-módszer gyakorlati alkalmazásának szép példája - a következő: tegyük fel, hogy adott egy bázismegoldás, ahol a bázisvektorok $P_j = /a_{1j}, a_{2j}, \dots, a_{mj}/^*$ - nyilván a bázisvektorok azok szabás-vektorok-, és a megfelelő anyag költsége c_j /j = 1, 2, ..., m/. Legyen $A_m \times m = /P_1, P_2, \dots, P_m/$ a bázisvektorokból álló mátrix. Ekkor a bázisba bevonandó ismeretlen $P = /a_1, a_2, \dots, a_m/$ *



vektor előáll, mint a bázisvektorok lineáris kombinációja /ahol U alkalmas m-dimenziós vektor/:

$$/3/ \quad A U = P$$

Csak olyan szabás-vektort érdemes és lehet figyelembe venni, amelynél ha c az alapanyag költsége, akkor

$$/4/ \quad C U > c, \text{ ahol } C = /c_{i_1}, c_{i_2}, \dots, c_{i_m}/.$$

Ha /3/-ből U-t kifejezzük és /4/-be helyettesítjük, megkapjuk a $C A^{-1} P > c$ egyenlőtlenséget - amely a szimplexmódszer-nél a P vektor bázisba vihetőségének feltétele -. Ez, ha a $C A^{-1}$ sorvektor elemeit $/\pi_1, \pi_2, \dots, \pi_m/$ -el jelöljük, akkor

$$/5/ \quad \pi_1 a_1 + \pi_2 a_2 + \dots + \pi_m a_m > c \text{ alakú lesz.}$$

Azt kaptuk, ha a $\pi_1 a_1 + \pi_2 a_2 + \dots + \pi_m a_m$ kifejezés értéke valamely egészértékű m-dimenziós pontban nagyobb, mint c, akkor a bázisba bevonhatjuk ezt a vektort, feltéve, hogy valamelyik szabáskép szabás-vektora.

Az előbbiek alapján nyertük a következő tételt /amely a szimplexmódszer speciális alkalmazását mutatja/:

2.1. Tétel: Legyen $A_{m \times m}$ egy lehetséges bázisa az /2/ LP-nek

$\pi_1, \pi_2, \dots, \pi_m$ árnyékköltségekkel és legyen $P = /a_1, a_2, \dots, a_m/$ c alapanyagköltségű szabás-vektor, amelyre

$$\sum_{i=1}^m \pi_i a_i > c.$$

Ekkor a P bázisba vitelével kisebb költségű bázis-
megoldást kapunk. Ha nincs olyan szabás-vektor,
amelyre ez az egyenlőtlenség teljesül, akkor az
optimális bázis $A_m \times m$.

Az /5/ egyenlőtlenség természetesen a szimplexmódszerből
adódik, tehát bármelyik LP-nél felírható, azonban nem mindig
adható meg nagyméretű feladat esetén a megfelelő vektor.
GILMORE és GOMORY felismerése azon az egyszerű tényen alapszik,
hogy a darabolási LP egy /5/-nek eleget tevő oszlopvektorát
generálni tudjuk.

Az /5/-nek eleget tevő szabás-vektor megkeresése egy
hátizsákfeladat felírását sugallja, amely a következő egy-
-feltételes diszkrét /itt egészértékű/ lineáris programozá-
si feladatot jelenti /lásd pl. [29] /:

$$x_j \geq 0, x_j = \text{egész } /j = 1, 2, \dots, m/$$

$$\sum_{j=1}^m a_j x_j \leq b \quad /a_j \geq 0, \gamma_j \geq 0; j = 1, 2, \dots, m/$$

$$\sum_{j=1}^m \gamma_j x_j \rightarrow \max!$$

Nyilvánvalóan alapvető, hogy a hátizsákfeladat megoldása
az valamelyik szabásminta szabás-vektora legyen, ezért a
hátizsákfeladat feltételének /ill. feltételeinek/ a dimenziót,
a darabolás műszaki körülményeit, jellemzőit kell kifejeznie.

Például: egy-dimenziós esetben, ha L a c költségű alapanyag hossza és l_1, l_2, \dots, l_m a kért rendelések hossza, akkor

$$/6/ \quad a_i \geq 0, a_i = \text{egész } /i = 1, 2, \dots, m/$$

$$\sum_{i=1}^m l_i a_i \leq L$$

$$\sum_{i=1}^m \pi_i a_i \rightarrow \max!$$

a megfelelő hátizsákfeladat.

Ha a hátizsákfeladat célfüggvényének optimális értéke nagyobb c -nél, akkor a megoldást /mint szabás-vektort/ bevihetjük a bázisba. Ha nem nagyobb c -nél, akkor az optimális szabásminták már a birtokunkban vannak, azaz már megoldottuk a feladatot.

Az /5/ egyenlőtlenségnek eleget tevő szabás-vektorok bármelyike bevihető a bázisba. Azonban GILMORE és GOMORY a hátizsákfeladat optimális megoldását javasolta, mivel ekkor kell - amint egy "ad hoc" algoritmussal összevetve kipróbálták - a legkevesebb bázistranszformációt elvégezni. Erre vonatkozóan részletes vizsgálatok futtatási eredményeit is közölték [34] .

A módszer természetesen nem önindító, így az indításhoz szükség van egy bázismegoldásra. Ez okozza általában a hasonló jellegű megközelítésekénél a problémát, mivel adott esetben

bekerülő szabás-vektorból /amelyet hátizsákgeladat megoldása-
ként kapunk/ a $P' = /-c, a_1, a_2, \dots, a_m/$ vektort /itt c az
alapanyag költsége/. Ekkor a G szimplex táblázat a következő
alaku lesz /ahol a $B_k^{-1} N'$ az aktuális megoldást mutatja a
 k -adik bázistranszformáció után/:

$$G_k = \left[\begin{array}{c|c|c} B_k^{-1} & & \\ \hline & B_k^{-1} N' & \\ \hline & & B_k^{-1} P' \end{array} \right]_{m+1 \times m+3}$$

A "szük keresztmetszet" elvét betartva - azaz a
 $B_k^{-1} N' / B_k^{-1} P'$ törtek közül a legkisebb tört nevezője lesz a
generálóelem - kiválasztjuk az utolsó oszlopból a generáló-
elemet és végrehajtjuk a bázistranszformációt. A transzformáció
után a táblázat /az új bázissal/:

$$G_{k+1} = \left[\begin{array}{c|c|c} B_{k+1}^{-1} & & \\ \hline & B_{k+1}^{-1} N' & \\ \hline & & e \end{array} \right]_{m+1 \times m+3}$$

ahol e megfelelő $m+1$ -dimenziós egységvektor.

A hátizsákgeladat optimális megoldása nem tartalmazhat
negatív számot, így segédváltozóhoz tartozó vektort sohasem
kapunk. Ebből következik, hogy a /2/ LP megoldása során - ha
csak a hátizsákgeladat megoldásaként kapott szabás-vektorokat
vesszük figyelembe - a táblázat nem-lehetséges megoldást is
mutathat. Ezt a $C A^{-1}$ vektor negatív értéke /vagy értékei/
jelzik, ilyenkor a megfelelő egységvektorok /amelynek nemzéró
eleme $-1/$ bázisba való bevitelével a megoldást megengedetté

tehetjük, azaz segédváltozóhoz tartozó vektort viszünk a bázisba.

Az LP fentiekben ismertetett nagyon praktikus, a darabolási feladathoz alkalmazkodó megoldása kis helyet igényel a számítógépben, mivel a bázisban levő szabás-vektorokon és az előbb bemutatott táblázaton kívül mást szinte nem is kell tárolni. A hátizsákfeladat megoldására pedig könyvtári programcsomag is felhasználható.

A szimplexmódszer degenerációjának elkerülésére perturbációs eljárást javasoltak [33], azonban ez ennél a gyakorlati feladatnál inkább csak elméleti jelentőséggel bír. Az elemi báziscsere-transzformációk számának és a felhasznált gépidőnek a csökkentésére azt is javasolták, hogy az adott megoldást optimálisnak fogadjuk el, ha a /2/ célfüggvényének értéke egy előre megadott korlátnál kevesebbet változik [76]. Ezzel a degenerációt is ki lehet szűrni.

A lineáris programozás, a módosított szimplexmódszer magyar nyelven több könyvben is megtalálható, többek között a [71] és [74] munkákban.

2.1.1. A darabolási feladat jellemző feltételei

Tegyük fel, hogy p -féle alapanyag található a raktárban, a k -adikból Q_k mennyiség $/k = 1, 2, \dots, p/$. A rendeléseket csak ezekből az alapanyagokból lehet kiszabni. Ekkor a /2/ LP-feladat a következő alakú lesz /bővül p darab feltétellel/:

$$x_j \geq 0 \quad /j = 1, 2, \dots, n/$$

$$\sum_{j=1}^n a_{ij} x_j \geq N_i \quad /i = 1, 2, \dots, m/$$

$$/7/ \quad \sum_{j=1}^n \delta_{kj} x_j \leq Q_k \quad /k = 1, 2, \dots, p/$$

$$\sum_{j=1}^n c_j x_j \rightarrow \min!$$

ahol $\delta_{kj} = \begin{cases} 1, & \text{ha a } j\text{-edik szabáskép a } k\text{-adik alapanyagból készült,} \\ 0, & \text{egyébként.} \end{cases}$

Amennyiben van egy induló bázismegoldásunk és a B_0^{-1} táblázatot is el tudjuk készíteni, akkor a módosított szimplexmódszert a /7/ feladatra a korábban leirt módon alkalmazzuk, mint-ha a rendelések száma $m+p$ lenne. Ekkor egy szabás-vektor meghatározásához p számú hátizsákfeladatot kell megoldani, pl. a k -edik alapanyagra a következő célfüggvény lesz /ahol π'_k a k -edik alapanyagra vonatkozó együttható, azaz a tábla első sorának $m+1+k$ -edik eleme/:

$$c_k - \sum_{i=1}^m \pi_i a_i + \pi'_k \rightarrow \min!$$

Mivel a π'_k az a_i -ktől nem függ, ezért olyan p darab háti-
zsákgeladatot kell megoldanunk, ahol a feltételek különböznek
és a célfüggvény azonos:

$$a_i \geq 0, a_i = \text{egész } /i = 1, 2, \dots, m/$$

$$\sum_{i=1}^m c_i a_i \leq L_k$$

$$\sum_{i=1}^m \pi_i a_i \rightarrow \max!$$

A komoly probléma akkor jelentkezik, amikor az induló bázis-
megoldást kell elkészítenünk, ami lényegében a kétfázisu szimplex-
módszer első lépését jelenti. Ha a korábban leírtak analógiá-
jára homogén szabásképek szabás-vektoraiból akarunk bázist ké-
szíteni, akkor bizonyos esetekben nehéz találni egy bázismeg-
oldást, másrészt nem biztos, hogy diagonális lesz a mátrix.
Ez akkor fordulhat elő, ha egy rendelést csak több különböző
alapanyagból lehet kiszabni, azaz ha sokféle, de kis darabszá-
mu tételekből áll a raktárkészlet $/p > m/$. A nagyméretű mátrix
inverze a kerekítések miatt ilyenkor a további számításokhoz
esetleg alkalmatlan lehet. Ezért a gyakorlatban a raktárkészle-
tet a GGM-nél vagy nem veszik figyelembe, vagy pedig heuriszt-
tikus megoldást adnak /pl. MARCONI [87] /. A 3. fejezetben
ismertetett heurisztikus eljárásnál és az 5. fejezetben leírt



megközelítésnél, amikor a "legelőnyösebb" szabás-vektorokat elkészítik és a nagyméretű ILP-t könyvtári programcsomaggal megoldják [79], a raktárkészlet feltétele egyszerűen csatolható a feladathoz. A 6. fejezetben bemutatunk egy olyan Kalmár Jánossal közösen készített heurisztikus eljárást, amely megad egy induló bázismegoldást [69].

Az alapanyagokra vonatkozó feltételek "erősségétől" függ a /7/ megoldásának jósága. Az igényelt rendelések összemennyiségének ismeretében célszerű megnézni, kellő rátartással megbecsülni azt, hogy a raktáron levő alapanyag elegendő? Amennyiben nem, akkor üres a lehetséges megoldások halmaza, nincs megoldása a feladatnak. Ha töröljük a raktárra vonatkozó feltételt, akkor termelésirányításra is lehetőség van. Ugyanis azt az alapanyagot rendeljük meg, gyártjuk le, amelyikre később a darabolás során szükség van. Ez utóbbi meggondolás feltételezi, hogy az optimalizálás és a tényleges darabolás között lényeges időbeli eltérés van. Másrészt az alapanyag minősége homogén és nem sérülékeny /pl. az üvegtábla eltörhet a darabolás közben is, ezért gyorsan kellene gyártatni a megfelelő alapanyagból/, illetve elegendő biztonsági készletet tárolunk.

A /7/ feladat adódik akkor is, amikor több, egymástól eltérő kapacitású szabásgép van és egy szabásgépen csak egyféle méretű alapanyag vágható fel /machine balance problem; [34] /. A cél ilyenkor a gépek kapacitásának kihasználása.

Gyakran előfordul egy másik speciális előírás is, amikor a megrendelőrendelések tételszámát alulról és felülről is korlátozza, azaz a tételszám bizonyos türeshatárok között mozoghat. Legyen N'_i az i -edik rendelés tételszámának az alsó, N''_i a felső korlátja $(N'_i < N_i < N''_i)$. Az ilyen megszorításokra javasolta GILMORE és GOMORY az alábbi hányados-programozási feladatot / 34 /:

$$x_j \geq 0 \quad /j = 1, 2, \dots, n, n+1, \dots, n+m/$$
$$/8/ \quad \sum_{j=1}^n a_{ij} x_j - x_{n+i} = N'_i \quad /i = 1, 2, \dots, m/$$

$$0 \leq x_{n+i} \leq /N''_i - N'_i/$$

$$\sum_{j=1}^n v_j x_j / \sum_{j=1}^n x_j \longrightarrow \min!$$

ahol v_j a j -edik szabáskép felületvesztesége /az alapanyag és a belőle kivágott rendelések felületének különbsége/. A célfüggvény a százalékos felületveszteség képletéből származik

$$100 \frac{\sum_{j=1}^n v_j x_j}{L} / \sum_{j=1}^n x_j$$

/ahol L az alapanyag felülete, mivel itt csak egy alapanyagot tételezünk fel/. Ez a megközelítés azért is jó, mivel az esetek döntő többségében a darabolás hatékonysági mutatójának a százalékos veszteséget tekintik. A /8/ feladat megoldására a [34] -ben ismertettek egy megoldást, amely a hiperbolikus programozás eredményeit használja fel.

Az alapanyagok mérete a veszteséget döntő mértékben befolyásolja. Akkor a legkedvezőbb, ha az alapanyagok mérete jóval nagyobb a rendelések méreteinél és többféle alapanyag van. Ezt ellensúlyozza az, hogy ilyenkor a gépidő nagy, mivel sokféle hátizsákfeladatot kell megoldani. Másrészt a veszteség alacsony, ha sok a rendelés, mivel a "jó" illeszkedésük valószínűsége megnő. Ekkor viszont a bázistranszformációk száma lesz magas és a hátizsákfeladatok nagyok lesznek, azaz szintén nő a gépidő. Ilyen jellegű vizsgálatokra közöl GILMORE és GOMORY [34] -ban futtatási eredményeket.

A darabolási feladat jellegéből adódik, hogy nemcsak az optimális megoldásra - a változók optimális értékeire - hanem a bázisvektorokra - az optimális szabás-vektorokra - is szükségünk van. A bázisba bevitt szabás-vektorokat a legcélszerűbb a bázistranszformáció elvégzésekor tárolni. A javasolt G táblázatban szerepel ugyan a bázisvektorokból álló mátrix inverze, így abból is meg lehetne kapni az optimális szabás-vektorokat, azonban az invertálás gyakorlati problémái miatt /mivel egészértékű elemekre van szükség viszonylag nagy mátrixnál/ ezt az utat nem tanácsoljuk.

A továbbiakban az egy- és két-dimenziós darabolásokat, a megfelelő hátizsákfeladatokat is a megoldásukra javasolt módszereket ismertetjük.

2.2. Az egy-dimenziós hátizsákfeladatok megoldási módszerei

A bázisba bevihető szabás-vektor meghatározását hátizsákfeladatra vezettük vissza. Ez a feladat egy-dimenziós esetben nagyon egyszerűen felírható. Legyen L az alapanyag, l_i az i -edik rendelés hossza $(i = 1, 2, \dots, m)$, ekkor a hátizsákfeladat /lásd /6//:

$$a_i \geq 0, a_i = \text{egész } (i = 1, 2, \dots, m)$$

$$\sum_{i=1}^m l_i a_i \leq L$$

$$\sum_{i=1}^m \pi_i a_i \rightarrow \max!$$

A feladathoz további feltételként a "vágófejek" számából adódó korlátozást is csatolhatjuk. Ugyanis a szabásgépek többségénél a vágófejek egyszerre vágnak, így legfeljebb annyi vágás végezhető egyidejűleg, annyi rendelés lehet egy szabásképben, ahány vágófej van. Két lehetőség kínálkozik, vagy a szabásképben szereplő rendelések számát kell korlátozni, vagy újra kell darabolni az alapanyag egy részét. Az ujradarabolásra a 3. fejezetben térünk ki.

Legyen a vágófejek száma: R . Ekkor a /6/ feladathoz csatolni kell a

$$/9/ \quad \sum_{i=1}^m a_i \leq R$$

feltételt /a jobboldal lehet $R-1$, amikor technológiai jellegű selejtcsikot vágunk az alapanyag mindkét szélén, és $R+1$, amikor egyik szélén sem vágunk/, azaz többfeltételes hátizsákfeladatot /ezt szokták hajórakodási feladatnak is nevezni/ kell megoldani. Előfordulhat másmilyen, például nem-lineáris feltétel is [79] :

$$/10/ \quad \sum_{i=1}^m \text{sign } a_i \leq S$$

amely azt fejezi ki, hogy a csomagolás, raktározás problémái miatt legfeljebb S különböző méretű rendelés lehet egy szabás-mintában.

Jellemző feltétel még egy szabásképben az egyes rendelések számának korlátozása, azaz a

$$/11/ \quad a_i \leq b_i \quad /i = 1, 2, \dots, m/$$

feltételek, amelyek automatikusan is előfordulnak, mivel $a_i \leq [L/ e_i]$.

A hátizsákfeladatot megoldó algoritmusok nagy része 0-1 típusu, azaz a változók csak 0 vagy 1 értékkel bírnak. Ha a_i a következő értékeket veheti fel,

$$a_i = 0, 1, 2, \dots, b_i$$

akkor felírhatjuk, hogy

$$a_i = \delta_1 + \delta_2 + \dots + \delta_{b_i}$$

$$\delta_j = 0 \text{ vagy } 1 \quad (j = 1, 2, \dots, b_i)$$

Azaz a következő lemmát kaptuk [72] :

2.2. Lemma: Véges sok értéket felvevő egészértékű változókat tartalmazó hátizsák feladat visszavezethető olyanra, amely csak 0-1 értékű változókat tartalmaz.

Természetesen a változók száma megnő, így az eljárások esetleg időigényesebbek lesznek.

A hátizsákfeladatok megoldására nagyon sok algoritmust /pl. [38] , [39] , [58] , [89] , [90] , [105] , [106] / sőt összefoglaló jellegű munkát is publikáltak [98] , a legtöbb számítógép programkönyvtárában van alkalmas program. A feladatot többek között KOVÁCS LÁSZLÓ BÉLA [72] , valamint FORGÓ FERENC [29] , [30] is ismerteti. Ennek ellenére vázoljuk a GILMORE és GOMORY által kidolgozott, a darabolási feladat jellegéhez alkalmazkodó eljárásokat, mivel a hátizsákfeladatok történetének szempontjából is érdekes, alapvető megközelítések, másrészt az egyik általánosításának tekinthetők az ún. "hátizsákfüggvényeket" felhasználó algoritmusok.

Az eljárások egy része a dinamikus programozáson alapszik, amelynek a BELLMAN által megfogalmazott általános elve a következő:

BELLMAN-féle optimalitási elv: Egy optimalitás döntéssorozat mindig rendelkezik azzal a tulajdonsággal, hogy bármilyen legyen is a kezdeti állapot és a kezdeti döntés, a fennmaradó döntések optimális döntéssorozatot alkotnak az első döntés eredményeképpen létrejött helyzetben.

Lényeges szempont, ha többféle alapanyag van, akkor a háti-
zsákfeladatokat szimultán oldjuk meg, azaz egyszerre, mivel
csak a feltételek jobb oldalában térnek el egymástól, GILMORE
és GOMORY a /6/ feladat megoldására [33] -ban a következő,
dinamikus programozáson alapuló, x-ben parametrikus eljárást
javasolta /elsősorban nem-lineáris feladatok esetén alkalmaz-
zák, lásd [30] /, ahol feltételezzük, hogy $L_1 > L_2 > \dots > L_p$:

$$0 \leq x \leq L_1$$

$$F_1 /x/ = \pi_1 \left[\frac{x}{e_1} \right]$$

⋮

$$F_s /x/ = \max_{0 \leq r = \left[\frac{x}{e_s} \right]} \{ r \cdot \pi_s + F_{s-1} /x - r \cdot e_s \}$$

/ahol [] az egészrész jele/.

Ekkor $F_m /L_1/$ kiszámolásával $F_m /L_2/, \dots, F_m /L_p/$ is automa-
tikusan adódik, tehát szimultán kell megoldani a p darab háti-
zsákfeladatot. Az eljárás hátránya az, hogy L_1 érték esetén a

a program tárigénye nagy, másrészt a rekurziós formula miatt az $F_m / L_1 /$ kiszámolása után "visszafelé" haladva kell a változók optimális értékét kiszámolni, ami időigényes.

Megjegyezzük, hogy többfeltételes hátizsákfeladat esetén az előbbi eljárás memóriaigénye a feltételek számával exponenciálisan nő [30]. A többfeltételes hátizsákfeladatot egyfeltételesé át lehet alakítani, azonban az így kapott feltétel jobb oldala, amitől a számításigényesség elsősorban függ, igen nagy lesz.

Ha az $F_s / x /$ -t úgy definiáljuk, mint az első s változó legjobb olyan összeállításának értékét, amely x hosszúságu alapanyagra készült, akkor adódik a következő dinamikus programozást felhasználó eljárás [35] :

$$0 \leq x \leq L_1$$

$$F_1 / x / = \pi_1 \left[\begin{array}{c} x \\ e_1 \end{array} \right]$$

⋮

$$F_s / x / = \max \{ \pi_s + F_s / x - e_s /, F_{s-1} / x / \}$$

$$/s = 2, 3, \dots, m/$$

Itt is $F_m / L_1 /$ kiszámításával együtt adódik $F_m / 2 /, \dots, F_m / L_p /$ értéke is. Az utóbbi eljárás nagy előnye az egyszerűbb képletben túl a változók értékének ügyes meghatározása, ugyanis az $F_s / x /$

függvényhez rendeljük egy $e^s /x/$ "backtrack"-függvényt, amely:

$$e^1 /x/ = \begin{cases} 1, & \text{ha } x \geq e_1, \\ 0, & \text{egyébként.} \end{cases}$$

⋮

$$e^s /x/ = \begin{cases} s, & \text{ha } F_s /x/ = \pi_s + F_{s-1} /x - e_s/ \\ e^{s-1} /x/, & \text{egyébként.} \end{cases}$$

$$/ s = 2, 3, \dots, m /$$

Ekkor például az L_1 hosszra nézve, ha $e^m /L_1/$ értéke r /és $r > 0/$ akkor a_r értéke legalább 1. Ezután nézzük $e^m /L_1 - e_r/$ értéket, ha ez r' /és $r' > 0, r = r'$ is lehet/, akkor a_r , értéke legalább 1 /illetve $r = r'$ esetén legalább 2/, stb. Nagyon lényeges, hogy az utóbbi eljárás memóriaigénye kicsi / $4 \cdot L_1$ értéknek kell helyet foglalni/ és a változók értékének meghatározása is nagyon gyors.

Ez a dinamikus programozást felhasználó, lényegében iteratív eljárás képezi az alapját a hátizsákfüggvények kiszámítását végző algoritmusoknak.

A fentieken kívül GILMORE és GOMORY adott egy olyan eljárást is, amely a vektorok lexikografikus rendezésén alapszik [34]. Ezt szintén sokat alkalmazzák, mivel rendkívül kicsi a memóriaigénye, csak annyi szabás-vektort kell tárolni, ahányféle alapanyag van. Az eljárást leírjuk részletesen, mivel a 6. fejezet-

ben ismertetett programrendszerben azt használjuk. Az algoritmus a lexikografikus rendezést felhasználva az összes szabásvektort leszámllálja, egyszerre figyelve minden alapanyagméretet. Az $a^1 = /a_1^1, a_2^1, \dots, a_{s_1}^1/$ vektor lexikografikusan nagyobb, mint az $a^2 = /a_1^2, a_2^2, \dots, a_{s_2}^2/$ vektor, ha van olyan i , amelyre $1 \leq i < \min /s_1, s_2/$ és $a_1^1 = a_1^2, \dots, a_i^1 = a_i^2$ teljesül, akkor $a_{i+1}^1 > a_{i+1}^2$. Rendezzük az alapanyagokat az $L_1 > L_2 > \dots$

$\dots \rightarrow L_p$, a változókat a $\frac{\pi_1}{e_1} \geq \frac{\pi_2}{e_2} \geq \dots \geq \frac{\pi_m}{e_m}$ sorrendbe.

A lexikografikusan legnagyobb vektorból kiidunlva - amely

$$a_1 = \left[\frac{L_1}{e_1} \right], \quad a_2 = \left[\frac{L_1 - a_1 e_1}{e_2} \right], \quad \dots, \quad a_m = \left[\frac{L_1 - \sum_{i=1}^{m-1} a_i e_i}{e_m} \right]$$

- megnézzük a lexikografikusan következő vektort, ahol a leszámllást az

$$\sum_{i=1}^s \pi_i a_i + \pi_{s+1} \frac{L_j - \sum_{i=1}^s e_i a_i}{e_{s+1}} > M'_j$$

egyenlőtlenséggel korlátozzuk /itt a lexikografikusan legnagyobb vektorhoz a célfüggvény értéke az L_j alapanyagban M'_j ; az s pedig olyan, hogy $a_j = 0$ ha $j > s/$.

Az algoritmus tevékenységi sémája:

$$1. \text{ lépés: } a_1 = [L_1 \ /e_1], \quad a_i = \left[\left(L_1 - \sum_{j=1}^{i-1} e_j a_j \right) / e_i \right] \ / i=2, 3, \dots, m/$$

$$t = 1 \text{ és } M_j = c_j \quad (j = 1, 2, \dots, p)$$

2. lépés. ha $\sum_{i=1}^m \pi_i a_i > M_j$ és $\sum_{i=1}^m e_i a_i \leq L_j$ akkor $M_j = \sum_{i=1}^m \pi_i a_i$

$$(j = t, t + 1, \dots, p)$$

$$s = \max \{i \mid 1 \leq i \leq m \wedge a_i \neq \emptyset\}$$

3. lépés. $a_s = a_s - 1$

$$t_1 = \min \{j \mid 1 \leq j \leq p \wedge \sum_{i=1}^s e_i a_i \leq L_j \wedge$$

$$(L_j - \sum_{i=1}^s e_i a_i) \pi_{s+1} > (M_j - \sum_{i=1}^s \pi_i a_i) e_{s+1}\}$$

$$t = \max \{\emptyset, t_1\}$$

ha $t > 0$ ugorj az 5. lépésre.

4. lépés. $a_{j+1} = (L_t - \sum_{i=1}^j e_i a_i) / e_{j+1} \quad /j = s, s+1, \dots, m-1/$
ugorj a 2. lépésre.

5. lépés. $s_1 = \max \{i \mid 1 \leq i \leq s-1 \wedge a_i \neq \emptyset\}$

$$s = \max \{\emptyset, s_1\}$$

6. lépés. ha $s > \emptyset$ ugorj a 3. lépésre

egyébként END.

Az algoritmus és az első futtatási tapasztalatok [34] -ban található meg. Az algoritmusba a vágófejek számának korlátozása /9/ egyszerűen beépíthető - az 1. és a 4. lépésben kell figyelem-

be venni -, míg a szabásmintában a rendelkezések számát korlátozó /10/ feltételt nem tudjuk figyelembe venni. Azért alapvető ez az eljárás, mert a darabolási feladat jellegéhez jól alkalmazkodik, a vágófejek korlátozását be lehet építeni, minden alapanyag-méretet egyszerre lehet figyelembe venni és a program helyigénye is roppant kicsiny / L_j -től független, 2 m p a tárigény/. Több programrendszerben ez az eljárás található meg /pl. [23], [61] /, az általunk készített, a 6. fejezetben leírt GGM egy lehetséges adaptációja is ezt az algoritmust alkalmazza.

GILMORE és GOMORY néhány egyszerű ötlettel meggyorsították az eljárást, többek között a következő triviális lemma felhasználásával:

2.3. Lemma: Ha $\pi_i \leq \pi_j$ és $e_i \geq e_j$ akkor $a_i = 0$.

A futtatásaik alapján ezzel a lemmával a hátizsákfeladatok változóit 30-ról átlagosan 18,2-re csökkentették, ami igen jelentős. Gyorsították még a GGM-t azzal is, hogy a rendelkezéseket a tétel számuk szerint két csoportra bontották, a hátizsákfeladatot csak az egyik csoportra oldották meg. Így egy szabásképben csak hasonló tétel számú rendelkezések szerepeltek, azaz egy bázisba kerülő szabás-vektort várhatóan sokszor kell alkalmazni. Ebből pedig az következik, hogy a /2/ célfüggvénye várhatóan lényegesen csökken. Ezt az ötletet "median method"-nak nevezték el.

Az előbbieken kívül az egy-dimenziós darabolás hátizsák-feladatának a megoldásához elsősorban INGARGIOLA és KORSH [58] , MARTELLO és TOTH [88] , [89] valamint GREENBERG és FELDMAN [38] , [39] eredményeit hasznosítják ill. ajánlhatók. Az egy-dimenziós darabolás háttéréről, a kapcsolódó elméleti kérdésekről GILMORE nyújt nagyon jó összefoglalást egyik munkájában 32 .

2.3. A két-dimenziós darabolási feladatok

A két-dimenziós "téglalapból-téglalapot-ortogonálisan" vágások közül GILMORE és GOMORY a műszaki gyakorlatban legnagyobb súlyt képviselő quillotine-vágásokkal foglalkozott /lásd 1/a. ábra/ [35] . Ekkor az alapanyag /alaptábla/ valamelyik oldalára merőlegesen széltől-szélig vágunk, azaz újabb két téglalapot hozunk létre, amelyeket szintén tovább kell vágni quillotine-módon, stb. Ez a vágási mód az egy-dimenziós feladatokra való visszavezetést, illetve iteratív, rekurzív algoritmusok alkalmazását teszi lehetővé. Az LP és a GGM a dimenzióktól független, így egyetlen, de igen nehéz problémával kell megbirkózni a bázisba bevihető szabás-vektor meghatározása során, két-dimenziós esetben a hátizsákfeladattal.

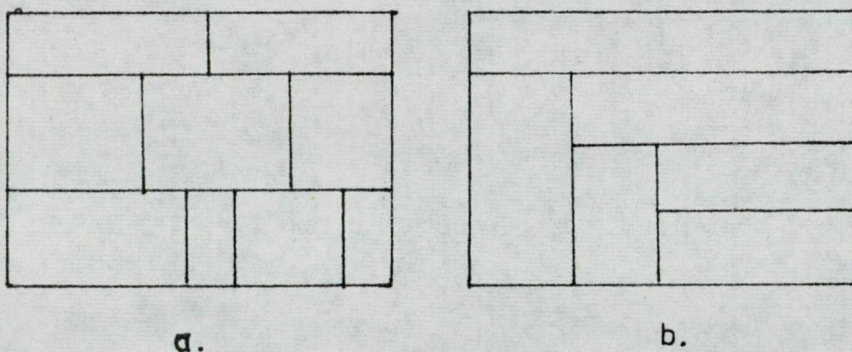
A két-dimenziós darabolásra a hátizsákfeladat verbálisan fogalmazva a következő: keressük az olyan $\{a_1, a_2, \dots, a_m\}^*$ nem-negatív, egészértékű vektort, amelyre a $\pi_1 a_1 + \pi_a a_2 + \dots + \pi_m a_m$

kifejezés felveszi a maximumát és amely vektor egy quillotine-vágás valamelyik szabásképének szabás-vektora.

A következő alpontban olyan két-dimenziós vágási módokat és megoldásukat ismertetjük, amelyek közvetlenül visszavezethetők egy-dimenziós feladatokra. Az általános két-dimenziós quillotine eset tárgyalására később térünk ki.

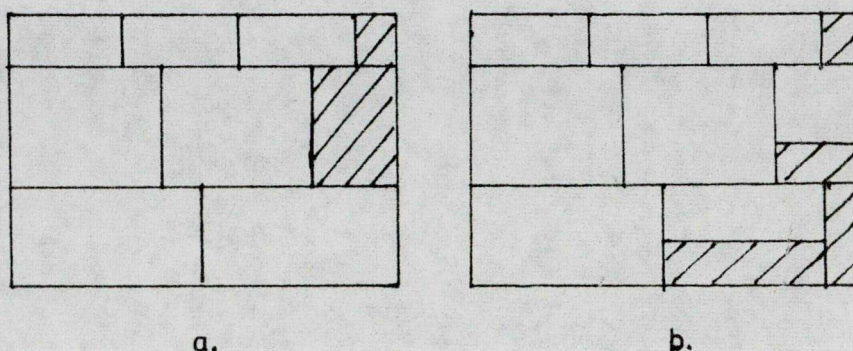
2.3.1. Két-ütemű quillotine-vágások

Az üvegtáblák, fémlemezek darabolásánál a leggyakrabban az alaptáblát valamelyik oldalával párhuzamosan csikokra vágják, majd ezeket a csikokat /mint egy-dimenziós alapanyagokat/ fel-
szabják a rendelt méretre, ill. ezeken a darabokon még egy vagy két további vágást végeznek el. Az ilyen jellegű quillotine-vágást két-üteműnek nevezzük /5a. ábra/. Az első ütemben a csikokat készítjük el, a második ütemben pedig a csikok egymás mellé illesztésével a legjobb szabásképet.



5. ábra. Két-ütemű /a/ és általános quillotine-vágás /b/ szabásképei

A két-ütemű vágások még további két csoportra oszthatók aszerint, hogy a csikoknál csak keresztvágást /6a. ábra/ vagy még további vágást is lehet végezni /6b. ábra/. Természetesen az utóbbi eset az általánosabb, így ezt ismertetjük.



6. ábra. Két-ütemű quillotine-vágások, ahol csak keresztvágás /a/, illetve még további vágás /b/ is elvégezhető.

A két-ütemű vágásra is a verbálisan megfogalmazott két-dimenziós hátizsákfeladat értényes, de itt az egy-dimenziós esetre való visszavezetés miatt egyszerű megoldási módszereket tudunk adni. Tegyük fel, hogy az alaptáblák hossza L_j és szélességük w_j / $j = 1, 2, \dots, P$ /, a rendelések méretei l_i és w_i / $i = 1, 2, \dots, m$ /. Rendezzük át a kért téglalapokat a $w_1 \leq w_2 \leq \dots \leq w_m$ -nek megfelelő sorrendbe. A bázisba bevihető szabás-vektor meghatározása kétütemben történik a j -edik alaptábla felhasználásával: az

első ütemben m db csikot adunk meg $/m$ db hátizsákfeladatot oldunk meg/, minden rendelés szélességhez egyet, a legjobbat. A második ütemben az m csik közül választjuk ki a legjobb szabás-képben szereplőket, a csik-szélességek illesztésével /amit újra egy-dimenziós hátizsákfeladat megoldása szolgáltat/:

/i/ a w_i szélességhez meghatározzuk a legjobb csikot, azaz a

$$a_k \geq 0, a_k = \text{egész } /k = 1, 2, \dots, i/$$

$$\sum_{k=1}^i \ell_k a_k \leq L_j$$

$$\sum_{k=1}^i \pi_k a_k \rightarrow \max!$$

hátizsákfeladatot megoldjuk. Legyen π_i^* a célfüggvény maximuma $/i = 1, 2, \dots, m/$.

/iii/ A csikok legjobb egymás mellé illesztését a következő hátizsákfeladat megoldása adja:

$$b_k \geq 0, b_k = \text{egész } /k = 1, 2, \dots, m/$$

$$\sum_{k=1}^m w_k b_k \leq W_j$$

$$\sum_{k=1}^m \pi_k^* b_k \rightarrow \max!$$

A bázisba bevihető szabás-vektort az előbb leirt két-ütemű vágás esetén $m+1$ darab egy-dimenziós hátizsákfeladat megoldásával határoztuk meg. Természetesen az első ütem csikjainak össze-

állítását meg kell őrizni, azaz melyik rendelésből hány szerepel bennük, mivel a második ütem után a szabás-vektort el kell készítenünk. Az előbb csak a rendelt téglalapok szélességét vettük a csikok szélességének. A legtöbb esetben azonban teljesen mindegy, hogy állítva vagy fektetve szerepel a rendelés valamelyik csikban. Ezt a legegyszerűbben fiktív rendelések közbeiktatásával érhetjük el. Legyen $w_{m+1} = l_i$ és $l_{m+i} = w_i$ / $i = 1, 2, \dots, m$ /, ahol most a négyzetektől eltekintünk. Így a $2m+1$ darab egydimenziós hátizsákfeladatot /ahol $2m$ a változók száma/ kell megoldanunk.

Amennyiben a csikokból kikerülő téglalapokat nem vághatjuk tovább /azaz csak a 6a. ábrán látható szabásképek megengedettek/, akkor az első ütem hátizsákfeladatain kell változtatni:

/i'/ A w_i szélességhez meghatározzuk a legjobb csikot, azaz a

$$a_k \geq 0, a_k = \text{egész} \quad /k = u, u+1, \dots, i/$$

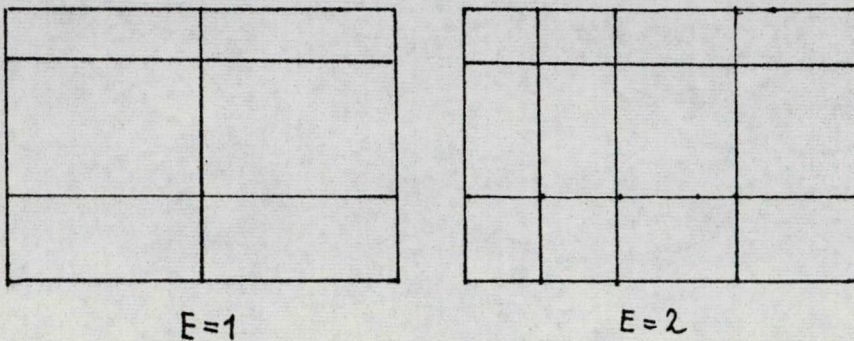
$$\sum_{k=u}^i l_k a_k \leq L_j$$

$$\sum_{k=u}^i a_k \rightarrow \max!$$

hátizsákfeladatot /ahol $w_u = \dots = w_i$ és $w_{u-1} \neq w_i$ / oldjuk meg. A célfüggvény maximuma legyen \prod_i^* .



A két-ütemű szabásképeket megkülönböztethetjük aszerint is, hogy a szabáskép hányféle csikot tartalmazhat. Jelölje a csikféleségek számát E . Egyes műszaki alkalmazásoknál, szabásgépeknél van ennek nagy jelentősége /7. ábra/, mivel a speciális feladatok az általános két-ütemű megközelítés alapján, de bizonyos egyszerűsítésekkel oldhatók meg.



7. ábra. Speciális két-ütemű szabásképek.

Ha E -re nem teszünk megszorítást, akkor két-ütemű szabad vágásról beszélünk. Ha az alapanyag több méretben van, akkor természetesen minden méretre a két-ütemű feladatot meg kell oldani. Egyszerűbb és gyorsabb a megoldás, ha a hossza vagy a szélessége minden alaptáblának egyenlő és a csikok vágása ezzel a fix mérettel párhuzamos. Ekkor a csikok minden alaptáblánál ugyanazok és így csak /ii/-t kell annyiszor megoldani, ahányféle alaptábla méret van.

2.3.2. Az általános quillotine-vágás

A két-dimenziós általános quillotine-vágás hátizsákfeladatának a meghatározása és megoldása kezdetben elméletileg is nehézségeket okozott. Erre a problémára GILMORE és GOMORY kifejlesztette az un. hátizsákfüggvények elméletét, amely alapján az egy- és két-dimenziós darabolás hátizsákfeladatának megoldására több algoritmust is megadtak [36].

Az egy-dimenziós esetben a hátizsákfeladat az alábbi /lásd /6//:

$$\begin{aligned} a_i &\geq 0, a_i = \text{egész } /i = 1, 2, \dots, m/ \\ \ell_1 a_1 + \ell_2 a_2 + \dots + \ell_m a_m &\leq L \\ \pi_1 a_1 + \pi_2 a_2 + \dots + \pi_m a_m &\longrightarrow \max! \end{aligned}$$

amiből egyszerű formalizálással kapjuk az egy-dimenziós hátizsákfüggvényt:

$$F(x) = \max \left\{ \pi_1 a_1 + \dots + \pi_m a_m; \ell_1 a_1 + \dots + \ell_m a_m \leq x, a_i \geq 0 \text{ és egész} \right\}$$

A cél a hátizsákfüggvény könnyen kiszámítható, algoritmizált alakban való megoldása. GILMORE és GOMORY először a két-dimenziós hátizsákfüggvényt határozta meg, amelynek speciális esete az egy-dimenziós.

A két-dimenziós hátizsákfüggvény verbálisan a következő:

$$F(x, y) = \max \left\{ \pi_1 a_1 + \pi_2 a_2 + \dots + \pi_m a_m; \right. \\ \left. (a_1, a_2, \dots, a_m)^* \text{ két-dimenziós} \right. \\ \left. \text{szabásvektor} \right\}$$

A hátizsákfüggvények kiterjedt elméletéből csak a darabolási feladatok szempontjából legfontosabb tételeket és algoritmusokat adjuk meg, a témakör bővebb kifejtése [36] -ban megtalálható.

A hátizsákfüggvényektől a következő alapvető tulajdonságokat várjuk el:

$$/i/ \quad F(x, y) \geq 0$$

$$/ii/ \quad F(x_1+x_2, y) \geq F(x_1, y) + F(x_2, y); \quad F(x, y_1+y_2) \geq F(x, y_1) + \\ + F(x, y_2)$$

$$/iii/ \quad F(\ell_i, w_i) \geq \pi_i \quad /i = 1, 2, \dots, m/$$

A fenti három feltételnek eleget tevő függvényeknek csak a részhalmaza hátizsákfüggvény:

2.4. Tétel: Az (ℓ_i, w_i) -méretű és π_i árnyékköltségű $/i=1, 2, \dots, m/$ hátizsákfeladatnak F akkor és csak akkor hátizsákfüggvénye, ha

a/ az $/i/ - /iii/$ feltételeknek eleget tesz, és

b/ az $/i/ - /iii/$ feltételeknek eleget tevő függvények között a legkisebb, azaz ha G eleget tesz

$/i/ - /iii/-$ nak, akkor

$$F /x,y/ \leq G /x,y/ \text{ minden } /x,y/-\text{ra.}$$

A hátizsákfüggvények kiszámítására a következő tételek által megoldott kifejezések lehetőséget adnak:

2.5. Tétel: A hátizsákfüggvény kielégíti a következő függvény-egyenletet:

$$/13/ \quad F /x,y/ = \max \left\{ F_0 /x,y/, F /x_1,y/ + F /x_2,y/, F /x,y_1/ + F /x,y_2/; \right. \\ \left. x \geq x_1 + x_2, 0 < x_1 \leq x_2, y \geq y_1 + y_2, 0 < y_1 \leq y_2 \right.$$

$$\text{ahol } F_0 /x,y/ = \max \left\{ 0, \pi_j; \ell_j \leq x \text{ és } w_j \leq y \right\}$$

A két-dimenziós tétel egy-dimenziós megfelelője a következő /amely a /12/ dinamikus programozáson alapszik/:

2.6. Tétel: Az egy-dimenziós hátizsákfüggvényt az alábbi egyenlet definiálja:

$$/14/ \quad F /x/ = \max \left\{ 0, F /x-\ell_i/ + \pi_i; i \geq \ell /x-\ell_i/, x \geq \ell_i \right\} \\ \text{ahol } \ell /x/ = \max \left\{ 1, i; F /x/ = F /x-\ell_i/ + \pi_i \right\}$$

A dinamikus programozást használó, a korábban ismertetett eljárások a következő lemmát veszik alapul [32] :

2.7. Lemma: Legyen $F /L/ = F /L-\ell_i/ + \pi_i$ és $F /L-\ell_i/ = F /L-\ell_i-\ell_j/ + \pi_j$

$$\text{Ekkor } F /L/ = F /L-\ell_j/ + \pi_j.$$

Bizonyítás: Definíció szerint $F / 0 / = 0$ és

$$F / L+1 / = \max \{ F / L+1 - \ell_1 / + \pi_1; L+1 \geq \ell_1 \text{ és } 0 \leq i \leq m \}$$

A lemma feltételeiből következik, hogy

$$F / L / \geq F / L - \ell_j / + \pi_j \quad \text{és} \quad F / L - \ell_j / - F / L - \ell_j - \ell_i / + \pi_i.$$

Következésképpen

$$F / L / \geq F / L - \ell_j / + \pi_j \geq F / L - \ell_j - \ell_i / + \pi_i + \pi_j$$

Igy

$$F / L - \ell_j - \ell_i / + \pi_i + \pi_j = F / L - \ell_i / + \pi_i = F / L /.$$

GILMORE és GOMORY négy eljárást adott az egy-dimenziós hátizsák-függvény kiszámítására. Ezek közül kettőt, az alapalgoritmust és a rendelések átrendezésén alapuló step-off algoritmust ismertetjük /ez utóbbira vonatkozó új eredményekre is kitérünk/.

A legáltalánosabb /időben is a leglassabb/ algoritmus, a "Basic step-off" a 2.5. tételben megadott /13/ kifejezés egy-dimenziós alakjának a kiszámítását végzi el, amely a következő:

$$F / x / = \max \{ F_0 / x /, F / x_1 / + F / x_2 /; x_1 + x_2 \leq x, 0 < x_1 \leq x_2 \}$$

$$\text{ahol } F_0 / x / = \max \{ 0, \pi_j; \ell_j \leq x \}$$

Egy-dimenziós step-off algoritmus

I. Legyen $F / x / = F_0 / x /$, $\ell / x / = x$ / $0 \leq x \leq L$ / , $x_2 = 1$

II.1. legyen $x_1 = 1$

2. ha $x_1 + x_2 > L$ menj II.4.

egyébként $V = F / x_1 / + F / x_2 /$

3. ha $V \leq F / x_1 + x_2 /$ menj II.4.

egyébként legyen $F / x_1 + x_2 / = V$, $e / x_1 + x_2 / = x_1$

4. ha $x_1 \geq x_2$ menj III.

egyébként $x_1 = x_1 + 1$ és menj II.2.

III. ha $x_2 = L$ END.

egyébként $x_2 = x_2 + 1$ és menj II.1.

Az előbbinél már gyorsabb a következő "ordered step-off" algoritmus.

Egy-dimenziós ordered step-off algoritmus

I. Rendezzük úgy sorba a paramétereket, hogy teljesüljön

$$\pi_1 / e_1 \leq \pi_2 / e_2 \leq \dots \leq \pi_m / e_m$$

legyen $F / x / = 0$ $/ 0 \leq x \leq L /$, $e / 0 / = 1$, $x_2 = 1$

II.1. legyen $j = e / x_2 /$

2. ha $x_2 + e_j > L$ menj II.4., egyébként $V = \pi_j + F / x_2 /$

3. ha $V < F / e_j + x_2 /$ menj II.4., egyébként $F / e_j + x_2 / = V$,

$$e / e_j + x_2 / = j$$

4. ha $j = m$ menj III.1., egyébként $j = j + 1$ és menj II.2.

III.1. ha $x_2 = L$ akkor END.

egyébként $x_2 = x_2 + 1$

2. ha $F / x_2 / > F / x_2 - 1 /$ menj II.1.

egyébként $F / x_2 / = F / x_2 - 1 / , \ell / x_2 / = m + 1$ és menj III.1.

Ezt az algoritmust módosította GREENBERG és FELDMAN egyik legújabb munkájában [39]. Ha két algoritmust összevetjük, akkor a GILMORE és GOMORY módszere az egyszerűbb szerkezetű, a cikluson belül kevesebb a műveletigény /nincs szükség segédfüggvények sorozatának a kiszámítására/, viszont a GREENBERG-FELDMAN algoritmus jelentősen redukálja a végrehajtandó ciklusok számát /ami különösen nagy L esetén előnyös/. A módosítás a II.2. lépésben történik, ahol

$$x_2 + \ell_j > L \text{ helyett az}$$

$x_2 + \ell_j > \lambda + \ell_m k_{\max}$ feltételt alkalmazzák /természetesen ez csak a lényegi változtatás, ami még maga után von néhányat/. A jelöléseket a módosítás alapjául szolgáló lemma ismertetése során vezetjük be. A lemma előtt a bevezető gondolatokra térünk ki.

Legyen a megoldandó feladat z maximalizálása, ahol

$$x_i \geq 0, x_i = \text{egész} /i = 1, 2, \dots, m/$$

$$\sum_{i=1}^m \pi_i x_i = z$$

$$\sum_{i=1}^m \ell_i x_i \leq L$$

Rendezzük át a változókat, hogy az indexek eleget tegyenek

a $\pi_m/e_m \geq \pi_i/e_i$ / $i \neq m$ / egyenlőtlenségnek. Ekkor $x_m \leq [L/e_m]$.

Ha az egyenlőség teljesül és L/e_m egész szám, akkor megoldottuk a feladatot, mivel $x_m = L/e_m$ és $x_i = 0$ / $i \neq m$ /. Egyébként vezessük be az $x_m = [L/e_m] - k$ új változót, így nyertük a következő feladatot z maximalizálására:

$$1' \quad \sum_{i=1}^{m-1} \pi_i x_i = z - \pi_m [L/e_m] + \pi_m k$$

$$2' \quad \sum_{i=1}^{m-1} e_i x_i \leq L - e_m [L/e_m] + e_m k$$

Ezzel a redukcióval egy könnyebben megoldható feladatot nyertünk, mivel a 2' jobb-oldala lényegesen kisebb lett. Minden k-ra $0 \leq k \leq [L/e_m]$ / meg kell oldani a feladatot és a legjobb megoldást kell kiválasztani.

Tegyük fel, hogy adott egy lehetséges megoldás: k^* és $x_1^*, x_2^*, \dots, x_{m-1}^*$ valamint z^* /pl. $k^* = 0, x_1^* = 0$ és $z^* = \pi_m [L/e_m]$ /. A cél olyan megoldás megkeresése, ahol $z > z^*$.

Olyan z-t keresünk, amelyre $z \geq z^* + d$, ahol

$$3' \quad d = \begin{cases} \text{l.n.k.o.} / \pi_1, \pi_2, \dots, \pi_m /, \text{ ha } \pi_i \text{ egész,} \\ \varepsilon > 0 \text{ /tetsz. kicsi/, egyébként.} \end{cases}$$

2.8. Lemma: A hátizsákfeladat $z \geq z^* + d$ megoldása létezésének feltétele rögzített z^*, k^* és x_i^* / $i = 1, 2, \dots, m-1$ /

mellett az, hogy a megoldásban K-ra teljesüljön

$$k \leq k_{\max}$$

4' ahol

$$k_{\max} = \left[\frac{\pi_r \lambda - e_r \delta}{b_r} \right]$$

és $\lambda = L - e_m \lfloor L/e_m \rfloor$; $\delta = z^* - \pi_m \lfloor L/e_m \rfloor + d$.

Az r index úgy van definiálva, hogy $\pi_r/b_r = \max \{ \pi_j/b_j / j \in I \}$
 ahol $I = \{ i | b_i \leq \pi_m L - e_m / z^* + d, 1 \leq i \leq m-1 \}$ és $b_i = \pi_m e_i - \pi_i e_m$.
 Ha I üres, akkor $\max z \leq z^*$.

Bizonyítás: A 2'-nek eleget tevő k és x_i értékekre, ha a
 $z \geq z^* + d$ -t behelyettesítjük az 1'-be, kapjuk, hogy

5'
$$k \leq \left(\sum_{i=1}^{m-1} \pi_i x_i + \pi_m \lfloor L/e_m \rfloor - z^* - d \right) / \pi_m$$

Hasonlóan 2'-ből nyerjük, hogy

6'
$$k \geq \left(\sum_{i=1}^{m-1} e_i x_i - L + e_m \lfloor L/e_m \rfloor \right) / e_m$$

Az 5' és 6' összevetéséből, valamint b_i definíciójából
 kapjuk, hogy

7'
$$\sum_{i=1}^{m-1} b_i x_i \leq \pi_m L - e_m / z^* + d$$

Ekkor a 7' megoldására adódik

$$x_r = \left(\pi_m L - e_m / z^* + d \right) / b_r; \quad x_i = 0 \quad / i \neq r /$$

A változók így kapott értékeit 5'-be helyettesítve, az egyszerűsítéseket elvégezve és a λ, δ jelöléseket bevezetve kapjuk a lemma állítását.

Az eljárás a szerzők állítása szerint különösen nagy L esetén hatékony. A tevékenységi séma a [39] -ben található meg. GILMORE és GOMORY a fentiekben ismertetett két eljárás mellett még másik kettőt megadott, amelyek a rendelkezések periodikusságát és a vágófejek /9/ feltételét veszik figyelembe.

KUUTTI és VOUTILAINEN a papirtekercsek feldarabolásakor MARTELLO és TOTH algoritmusát alkalmazták. Szerintük a darabolási feladathoz ez az algoritmus illeszkedik a legjobban, amelyet a vágófejek korlátozásával is kiegészítettek. Az algoritmus a [88] munkában található, itt nem ismertetjük.

A két-dimenziós hátizsákfeladatok kiszámítására két algoritmust adott GILMORE és GOMORY, amely a /13/ függvényegyenletet realizálja. Az általánosabb a két-dimenziós step-off algoritmus, amely az alábbi:

Két-dimenziós step-off algoritmus

- I. Legyen $F(x, y) = F_0(x, y)$; $0 \leq x \leq L$ és $0 \leq y \leq W$
és $l_i, w_i = l_i, w_i$ ($i = 1, 2, \dots, m$)
 $l_i, w_i = 0, x_2 = y_2 = 1$.

II.1. $x_1 = 1$

2. ha $x_1 + x_2 > L$ akkor menj III.1.
egyébként $V = F(x_1, y_2) + F(x_2, y_2)$.

3. ha $V > F/x_1 + x_2, y_2/$ akkor $F/x_1 + x_2, y_2/ = V,$
 $l/x_1 + x_2, y_2/ = x_1, w/x_1 + x_2, y_2/ = y_2$ menj II.4.

egyébként ha $V < F/x_1 + x_2, y_2/$ menj II.4.

egyébként $l/x_1 + x_2, y_2/ = x_1.$

4. ha $x_1 \geq x_2$ menj III.1.

egyébként $x_1 = x_1 + 1$ menj II.2.

III.1. $y_1 = 1$

2. ha $y_1 + y_2 > W$ menj IV.1.

egyébként $V = F/x_2, y_1/ + F/x_2, y_2/.$

3. Ha $V > F/x_2, y_1 + y_2/$ akkor $F/x_2, y_1 + y_2/ = V, l/x_2, y_1 + y_2/ = x_2$
 $w/x_2, y_1 + y_2/ = y_1$ menj III.4.

egyébként ha $V < F/x_2, y_1 + y_2/$ menj III.4.

egyébként $w/x_2, y_1 + y_2/ = y_1.$

4. ha $y_1 \geq y_2$ menj IV.1.

egyébként $y_1 = y_1 + 1$ menj III.2.

IV.1. ha $x_2 = L$ menj IV.2.

egyébként $x_2 = x_2 + 1$ menj II.1.

2. ha $y_2 = W$ STOP.

egyébként $y_2 = y_2 + 1, x_2 = 1$ menj II.1.

A két-dimenziós algoritmusok hely- és időigényesek, a gyakorlatban azért is alkalmazzák ritkán, mivel a két-ütemű quillotine-vágás a leggyakoribb az üzemekben és nem az általános



quillotine-vágás. Megjegyezzük, hogy a két-dimenziós darabolás hátizsákeladataira HERZ valamint GHRISTOFIDES és WHITLOCK az előbbieken említetteknel hatékonyabb eljárásokat készítettek. HERZ egy olyan rekurzív algoritmust adott meg, amely az ismétlődő vágásokat figyelembe véve lényegesen kevesebb memóriát igényel és elég gyors [53]. CHRISTOFIDES és WHITLOCK az $a_i \leq b_i$ / $i = 1, 2, \dots, m$ / feltételt /azaz egy rendelés számát szabásmintán belül korlátozták/ is beépítették fabejáráson alapuló leszámlálási algoritmusukba [12]. A felesleges ismétlődések elkerülésére szintén /HERZ-hez hasonlóan/ kiszűrték a "szimmetrikus" alakzatokat. A hátizsákeladatok elméleti kiterjesztésével, megoldási módszereivel a 0-1 típusú feladatok esetén többek között WEINGARTNER és NESS is foglalkozott [108].

Olyan algoritmussal, amely általános nem-quillotine-vágás szabás-vektorát készíti el - egy téglalap lefedését nem-átfedő téglalapokkal - nem talákoztunk. De viszont ALBANO és ORSINI eredményeit [7], amelyek szabálytalan alakzatok kiszabására készültek, fel lehetne használni erre a feladatra.

2.4. Javaslatok a GGM-hez

A következőkben két olyan egyszerű javaslatot ismertetünk, amelyek közvetlenül kapcsolódnak GILMORE és GOMORY munkáihoz. Mindkettő az optimális szabásminták számának a csökkentésére törekszik.

Egészértékű megoldást úgy kapunk, ha az LP optimális megoldását felfelé kerekítjük. Ennek a kerekítésnek a hibája kiszámu rendelés esetén jelentős lehet, például egy-dimenziós esetben ha 3 db 90 egységnyi és 3 db 100 egységnyi rendelést kell 200 egységnyi alapanyagból kiszabni, akkor a GGM által szolgáltatott megoldás [49] :

szabásképe 1	szabásképe 2
2 100	0 100
0 90	2 90
$x_1 = 1,5$	$x_2 = 1,5$

holott a következő megoldás

szabásképe 1
1 100
1 90
$x_1 = 3$

egy alapanyaggal kevesebbet használ fel.

HAESSLER azt az első ránézésre meglepőnek tűnő javaslatot tette, hogy egy szabásképen belül a rendelések számát korlátozzuk [49], ezáltal szerinte az optimális megoldás kevesebb szabásképből áll majd és így a kerekítésekből származó tulszabás is kevesebb lesz. Egy-dimenziós eseteket vizsgált, a módosított szimplexmódszer induló megoldásánál és a hátizsákfeladatoknál is az $a_i \leq b_i$ / $i=1,2, \dots, m$ / feltételt figyelembe vette /ahol a b_i problémától függő felső korlát/. HAESSLER felvetései nem

ujak /pl. CHRISTOFIDES és WHITLOCK [12] és KUUTTI-VOUTILAINEN [76] is alkalmazza/, inkább az a szokatlan, amilyen célból javasolta őket. Mindenesetre a próbafuttatások GGM-el való összevetései őt igazolták, átlagosan 5,5 %-kal csökkent a szabásképek száma és 49 %-kal a tulszabásoké /93 tekercs helyett csak 56 lett a többlet/. Természetesen a számításokhoz szükséges gépidő /a bázistranszformációk számának emelkedése miatt/ megnőtt.

A másik módosítást KUUTTI és VOUTILAINEN javasolta [76]. Ők szintén arra törekedtek, hogy az optimális megoldás minél kevesebb szabásképből álljon - matematikai értelemben az alternatív optimumok közül kell a legelőnyösebbet kiválasztani. Kiindulásuk az, hogy fiktív rendeléseket hoznak létre kettő vagy több rendelés összevonásából. Például egy-dimenziós esetben legyenek a rendelések:

	hossza	tételszáma
	100	50
	30	80
	45	80
ekkor az utolsó kettőt összevonva redukáljuk két rendeléssé		
a hármat,	100	50
	75 /30+45/	80

Természetesen a tételszámoktól erősen függ, hogy a rendelések számát mennyire lehet csökkenteni. A továbbiakban ők is a

GGM-t alkalmazzák. Megfigyeléseik szerint az összes gépidő 80 %-t a hátizsákfeladatra kell fordítani. Ők a MARTELLO-TOTH algoritmust tartották a legalkalmasabbnak [83], amelyet a vágófejek korlátozásával kibővítettek. A gépidőt jelentősen csökkentették azáltal is, hogy a futtatást befejezték, ha a /2/ lineáris programozási feladat célfüggvényének értéke egy előirt korlátnál kevesebbet változott. Azaz nem a minimális alapanyag-költségű megoldást keresték, hanem a felületvesztesség, az optimális szabásképek száma és felhasználása /azaz egy szabásképet elég sokszor kelljen alkalmazni/ közötti egyensúlyra törekedtek.

2.5. A GGM alkalmazásai

A következőkben röviden ismertetünk néhány olyan dolgot, amely a GGM alkalmazását mutatja be. Az egy-dimenziós esetekre - de a két-dimenziósra is - még a 60-as években több programrendszert készítettek, amit széleskörben használtak ill. használnak ma is /JOHNSTON hat ilyen rendszerre hivatkozik [67] /. JOHNSTON és BOURKE beszámol egy IBM program módosításáról, amely papirtekercsek felvágásához készült [61]. Az eredeti program a rendeléseket különböző szabásgépekre tudta elosztani, feltéve, hogy bármelyik rendelést bármelyik gépen le lehet vágni. A hátizsákfeladatot a korábban vázolt lexikografikus rendezésen alapuló eljárással oldották meg [34]. A gyakorlati alkalmazás során felvetődött igények kielégítésére néhány módosításra volt

szükség, amellyel a rendszer szolgáltatásait bővítették:

- lehetővé vált a rendelések korlátozása bizonyos gépekre,
- a rendeléseket egy szabásképben korlátozni lehet
$$/a_i \leq b_i, i = 1, 2, \dots, m/$$
- egy adott rendelés költségének szabásgéptől való füg-
gését,
- és a rendelések tételszámának alulról és felülről való
korlátozását engedélyezni lehet.

Az első három módosítást a hátizsákfeladatot megoldó eljárás-
ba építették be, a rendszer egyszerre legfeljebb 50 rendelés
felvágásának optimalizálását tudja elvégezni. A korábbi kézi
programozáshoz képest 12 %-kal nőtt a termelékenység és 0,5 %-
kal csökkent a hulladék /amely addig is elenyésző volt/.

A két-dimenziós darabolási feladatok közül elsősorban a
két-üteműekre irtak programokat. MARCONI szintén egy program-
rendszer jelentős átdolgozását mutatja be, nagyszámu futtatási
eredményt mellékelve [87]. Elsősorban a kések számának a
veszteségre, a bázistranszformációk számára és a gépidőre gya-
korolt hatását vizsgálta. Az általa megadott rendelésállományok
esetében a kések számát növelve csökkent a felületveszteség és
a bázistranszformációk száma, míg a gépidő lényegesen nem vál-
tozott.

DYSON és GREGORY üvegtáblák két-ütemű felszabását írják le [23]. A hátizsákfeladatok megoldására ők is a lexikografikus eljárást használták [34]. Az alkalmazás során felvetődött az ún. "folyamatos leszabás" problémája, azaz ha egy rendelés szerepel egy optimális szabásképben, akkor addig szerepeljen a soron következőkben, amíg ki nincs elégítve. A feladat ilyenkor az optimális szabásképek olyan sorrendjének megadása, amelyben a legtöbb rendelés folyamatosan lesz leszabva. Ez a követelmény a rendelések csomagolásának, raktározásának nehézségei miatt jelentkezett /lásd [79] /. DYSON és GREGORY az optimális szabásképek "legelőnyösebb" sorrendjét az "utazó ügynök" problémára való visszavezetéssel kereste meg. Azonban 20 rendelésnél /ill. szabásképnél/ többet nem tudtak figyelembe venni a gépidő nagyarányu megnövekedése miatt. Leírtak egy heurisztikus eljárást is, amely a felületveszteséget és a folyamatosságot együtt vizsgálja. Ennek lényege, hogy a legkisebb veszteségű szabásképet elkészítik és annyiszor leszabják, ahányszor lehet, majd elkészítik a maradék rendelésállományra a folyamatosságnak elegendő legkisebb veszteségű szabásképet, stb. Sajnos a végére a felületveszteség megnő. Ezt az előbb vázolt rendezési feladatot lehet kissé általánosítani, ugyanis előfordulhat, hogy az optimális szabásminták olyan legelőnyösebb sorrendjét kell megadni, ahol a rendelések vagy az egy megrendelőhöz tartozó rendelések folyamatossága, a sürgősségi szempontok, a határidők, stb. határozzák meg a "prioritást". Az üvegszabáshoz kapcsolódó másik

igen gyakori problémával foglalkozott CHAMBERS és DYSON [11] . A raktárban adott méretű és darabszámu félkésztermékek vannak, amikből a rendeléseket ki kell szabni. A rendelések ismerete nélkül meg kellene adni az üvegtáblát gyártóknak, hogy milyen méretű alaptáblát lehetne később a legelőnyösebben felhasználni. Természetesen figyelembe kell venni, hogy a megrendelők nagy része és a nyilászárók mérete egy éven belül nem változik lényegesen. A feladat matematikai vetületével foglalkozott WOLFSON [109] és SHAPIRO [101] , egyik speciális esetével PAGE [93] , míg HINXMAN a vonatkozó irodalmat is áttekinti már hivatkozott művében [56] .

MADSEN szintén üvegtáblák két-ütemű feldarabolását mutatja be egy kisüzem körülményeire adaptálva, ahol sokféle, de kis tételszámu rendelést kell kiszabni [83] . A hátizsákfeladatot hátizsákfüggvényt realizáló, a rendelések rendezésén alapuló algoritmussal oldotta meg [36] . Itt is felvetődött az egy megrendelőhöz tartozó rendeléseknek /ill. rendelésnek/ a folyamatos leszabása. MADSEN az optimális szabásképek sorrendjének meghatározását egy szimmetrikus bináris mátrix főátlója sáv szélességének minimalizálására /bandwidth problem/ vezette vissza [83] , [84] . Definiáljuk a következőképpen a szimmetrikus A mátrixot:

$$a_{ij} = \begin{cases} 1, & \text{ha az } i\text{-edik és a } j\text{-edik szabásképpen} \\ & \text{szerepel közös rendelés } /i \neq j/, \\ 0, & \text{egyébként.} \end{cases}$$

A sávszélességet / A / a következőképpen határozzuk meg:

$$\beta(A) = \max_i /i - \min_j /j \mid a_{ij} \neq 0/ /$$

A feladat $\beta(A)$ minimalizálása sor- és oszlopcserékkel, amelyre MADSEN a CUTHILL-McKEE algoritmus egy módosítását használta [19]. Egy másik megközelítést is alkalmazott, amikor a folyamatosságot az utazó ügynök problémára vezette vissza [86]. Az optimális szabásképek átrendezésének elméleti hátterével VOUTILAINEN foglalkozott részletesen [107], ill. FALEY és RICHARDSON [27].

A fentiekben ismertettük a darabolási feladatok számítógépes megoldására legszélesebb körben használható - és önmagában is szép - módszer alapjait és néhány alkalmazását. Törekedtünk a gyakorlati felhasználás során legtöbbször felvetődő kérdések tisztázására is. Ott ajánljuk a GGM-t, ahol a darabolási probléma lineáris programozási feladatként megfogalmazható, az egészértékre való kerekítés nem okoz gondot, valamint a bázisba bevonandó szabás-vektorokat hatékonyan és gyorsan elő lehet állítani.

A következő fejezetben a HAESSLER-féle heurisztikus eljárást ismertetjük, amely egy-dimenziós darabolásoknál alkalmazható.

3. Egy-dimenziós darabolási feladat nem-lineáris célfüggvény esetén

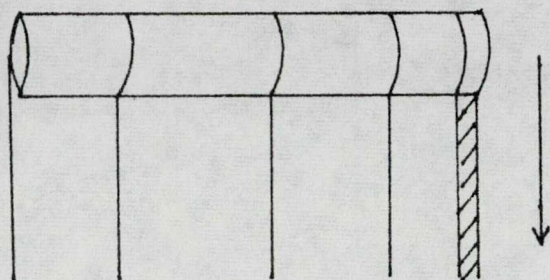
Az alábbi fejezetben olyan darabolási feladatokat - és a megoldásukra javasolt módszereket - ismertetünk, amelyeknél a célfüggvény nem-lineáris. Ebben a problémakörben HAESSLER és JOHNSTON végezte el az alapvető munkákat.

A 60-as években GGM-t realizáló programcsomagot készített több nagy software-gyártással is foglalkozó cég /már hivatkoztunk a [67] -ben levő felsorolásra/. Egy-egy ilyen univerzális program a darabolás egyedi, az adott üzem konkrét műszaki feltételeinek nem mindig tudott eleget tenni /pl. a 2.5.-ben már ismertetett JOHNSTON-BOURKE módosításra [61] ezért volt szükség/, mivel a GGM elsősorban az alapanyag költségének minimalizálására képes és a költségek nemcsak az alapanyagtól függnnek. A 70-es években előtérbe kerültek olyan módszerek, amelyek az egy-dimenziós feladatoknál a költségek eloszlását jobban tudják követni. A fejezetben ilyen jellegű eljárásokat ismertetünk egy-dimenziós esetben, amelyek elsősorban papirtekercek feldarabolásához alkalmazhatók. Mikor lép fel a nem-lineáris célfüggvény? Például:

- Amennyiben a kések száma kevés, akkor a szabáskép készítésénél ez korlátozó feltétel, vagy az alapanyag egy részét újra kell darabolni. Az ujradarabolás /papirteker-

cseknél az újratekercselés/ azonban jelentős többletköltséget és időráfordítást is igényel. Ilyenkor a felületveszteség és az újradarabolás együttes költségének minimumát keressük.

- A szabásminták váltása is újabb ráfordítást igényel /pl. a kések átállítása munka- és időigényes/, így ezt is figyelembe kell venni /azaz az optimális szabásminták számát csökkenteni kell/.



8. ábra. Tekercsek feldarabolása

A következőkben előbb két darabolási feladatot ismertettünk, majd azt követően a megoldásukra javasolt módszert.

3.1. A nem-lineáris darabolási feladatok

HAESSLER 1971-ben publikálta [40] az egy-dimenziós darabolási feladatot nem-lineáris célfüggvénnyel és a megoldására javasolt eljárást, amelynek több speciális alkalmazását más munkáiban ismertette. A feladat - amely papirtekercsek feldarabolására készült - megfogalmazása és formalizálása előtt

a gyártás, darabolás olyan körülményeire térünk ki röviden, amit figyelembe kell venni a modellezésnél. A papirgyártó gép kapacitása adott és folyamatosan dolgozik, tehát nem állhat meg. A papir onnan a tekercselőgéphez kerül, ahol a darabolás is megtörténik. A szükséges ujradarabolásokat egy "ujratekercselő" gépen végzik el. Mikor használják az utóbbit? Egyrészt, ha a kések száma kevés a tekercselőgépnél. Másrészt a papirgyártó és a tekercselőgép között tárolásra nincs sok lehetőség, valamint az átállítás az új szabásmintára /a kések átállítása/ időigényes, ezért ha egy szabásképet legfeljebb 3 alapanyagra /alapterkercsre/ kell alkalmazni, akkor a tekercselőgép kb. középen kettévágja az alapterkercset és küldi az ujraterkercselőhöz. Lényeges, hogy egy kést /másik kettő mindig rögzített és az egyenetlen szélet vágja le/ kell beállítani, ami gyorsan megy. Amennyiben 4 alapterkercset kell felválni, akkor az alapterkercs egyeik felét a rendelt méretekre szabja a tekercselő, a másikat át küldi az ujraterkercselőhöz. Legalább 5 alapanyag feldarabolása esetén a tekercselőgép a szabásmintának megfelelően vág /amennyiben a kések száma ezt lehetővé teszi/.

Az előbbi üzemi feltételeket kissé általánosítva kapjuk a következő feladatot: legyen m a rendelések száma, W_i a szélessége, R_i a tételszáma / $i = 1, 2, \dots, m$ / és az alapanyag szélessége W . A szabásképek két típusát különböztetjük meg aszerint, hogy a kések száma kevés / $k=2$ / és elég a tekercselőgépen / $k=1$ /, ahol k az indexpár első tagját jelöli.

$$x_{1j}, x_{2j} \geq 0, \text{ egészek}$$

$$/15/ \quad R_e \leq \sum_{j \in s_1} A_{1j} x_{1j} + \sum_{j \in s_2} A_{2j} x_{2j} \leq R_u$$

$$C_t \left(\sum_{j \in s_1} T_{1j} x_{1j} + \sum_{j \in s_2} T_{2j} x_{2j} \right) + C_r \left[\sum_{j \in s_1} x_{1j} (\delta(x_{1j} - K_p) + \delta(x_{1j} - K_c)) + \sum_{j \in s_2} x_{2j} (\delta(x_{2j} - K_p)) \right] / \left(\sum_{j \in s_1} x_{1j} + \sum_{j \in s_2} x_{2j} \right) \longrightarrow \min!$$

ahol x_{1j} a $k=1$, x_{2j} a $k=2$ típusu j -edik szabásképhez tartozik,

s_1 a $k=1$ típusu szabásképek halmaza,

s_2 a $k=2$ típusu szabásképek halmaza,

R_e és R_u m -dimenziós oszlopvektorok, amelyek i -edik eleme az i -edik rendelés tételszámának alsó ill. felső korlátja,

A_{kj} a k -típusu j -edik szabáskép szabás-vektora,

a_{ikj} az adott k -típusu j -edik szabásképben az i -edik rendelés száma,

C_t a hulladék költsége egységenként,

C_r az újratekerceselés költsége tekercsenként,

$$\delta / z-b/ = \begin{cases} 1, & \text{ha } 0 > z-b > -b \\ 0, & \text{egyébként} \end{cases}$$

K_c azt jelöli, hogy egy szabásképet ennyiszor kell legalább alkalmazni ahhoz, hogy a tekercselőgép teljesen felvágja /a konkrét példánál $K_c=5$ /,

K_p azt mutatja, hogy a tekercselőgép a szabásképet ennyiszor alkalmazva tudja még a tekercs felét a rendeléseknek megfelelően felvágni / $K_p=4$ /,

$$T_{kj} = W - \sum_{i=1}^m A_{ikj} W_i \quad \text{a leeső hulladék szélessége.}$$

A modell feltételei a rendelések tételszámait korlátozzák, azaz bizonyos türéshatárok között mozoghat a darabszám. A célfüggvény első ránézésre bonyolultnak tűnik. Az összeg első tagja a hulladék költségét mutatja a két szabásképtípusnál. A második tag az újratekercselések költségét tartalmazza az összes tekercshez viszonyítva. Azt fejezi ki, ha a $k=1$ típusu szabásképnél $x_{1j} < K_p$, akkor félbevágja a tekercselőgép az alaptekercset, így $2x_{1j}$ darabot kell újratekercselni. Ha $K_p \leq x_{1j} < K_c$, akkor x_{1j} -szer kell újratekercselni, mivel az egyik felét az alaptekercsnek már a rendelt méretekre vágjuk. Hasonlóan $K=2$ esetén, ha $x_{2j} < K_p$ akkor $2 x_{2j}$ -szer kell újratekercselni. Egyébként x_{2j} -szer, mivel a kések száma a szabáskép feldarabolásához kevés.

A másik feladat jóval egyszerűbben írható fel, a szabásképek váltásának /a gépnél a kések átállításának/ költségét tartalmazza és az újratekercselést nem vesszük figyelembe. Ezzel a feladattal HAESSLER a [41] dolgozatban foglalkozik /a jelölések egy része az előbbi feladatban is szerepel/:

$$\begin{aligned} & X_j \geq 0, X_j = \text{egész} \\ /16/ \quad R_l & \leq \sum_{j \in S} A_j X_j \leq R_u \\ C_t & \sum_{j \in S} T_j X_j + C_c \sum_{j \in S} \delta /X_j/ \rightarrow \min! \end{aligned}$$

ahol S a lehetséges szabásképek halmaza,

C_c a szabáskép váltásának költsége,

$$\delta /X_j/ = \begin{cases} 1, & \text{ha } X_j > 0, \\ 0, & \text{egyébként.} \end{cases}$$

A modell feltételében itt is a rendelés tételszámát korlátozzuk. Mivel újratekerccselés nincs, ezért a célfüggvény csak a hulladék és a szabásképek váltásának költségét tartalmazza.

3.2. A szekvenciális heurisztikus eljárás

Az előbb ismertetett nagyméretű egészértékű nem-lineáris feladatokat egzakt módszerrel megoldani gyakorlatilag lehetetlen, ezért mindkét feladat megoldásához MAESSLER az általa kifejlesztett szekvenciális heurisztikus eljárást használta. Az alapötlet egyszerű, elkészítünk egy "jó" szabásképet, azt ahányszor lehet alkalmazzuk, azután a megmaradt rendelésekre ismét készítünk egy "jó" szabásképet, stb., amíg minden rendelést tételszámának megfelelően ki nem elégítünk. Ezt "kiüritő" eljárásnak is szokták nevezni. Rögtön felvetődik, hogy az eljárás végére a nehezen illeszthető rendelések maradnak, amelyek csak nagy veszteség árán szabhatók ki. Ezt elkerülendő, HAESSLER a maradék rendelésállomány elemzésénél, a jó szabáskép - amely

a későbbieket nem rontja el - elkészítésénél két jellemzőt vett figyelembe:

/i/ a maradék rendelések kielégítéséhez szükséges alapteker-
csek becslését:

$$\sum_i R'_i W_i / W$$

/ahol R'_i az i-edik rendelésből még hátralévő darabszám/,

/ii/ és a szabásképben a rendelések átlagos számát:

$$W \sum_i R'_i / \sum_i R'_i W_i$$

/amely a $\sum_i R'_i / [(\sum_i R'_i W_i) / W]$ kifejezésből származik/.

A fenti becslések alapján három feltétel, a veszteség, a szabásképben szereplő tekercsek száma és a szabáskép felhasználásának száma által meghatározott "elfogadási szintet" vett figyelembe:

1. A maximális megengedhető veszteség MAXTL, azaz

$$W - \sum_i A_{ij} W_i \leq \text{MAXTL}$$

/ahol A_{ij} az i-edik rendelés száma a j-edik szabásképben/.

2. A szabásképben a tekercsek minimális és maximális száma MINR és MAXR,

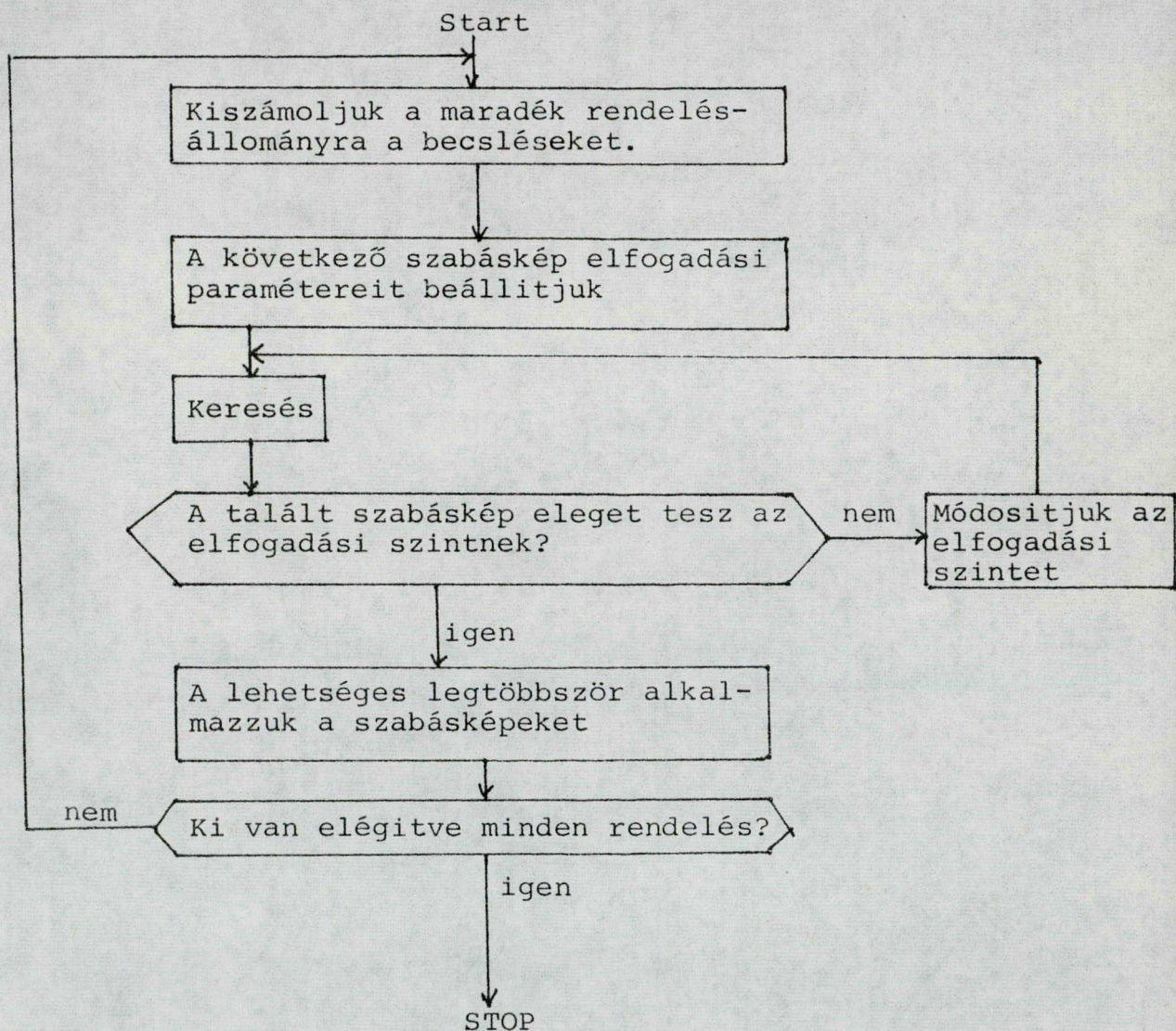
$$\text{tehát} \quad \text{MINR} \leq \sum_i A_{ij} \leq \text{MAXR}.$$

3. A szabáskép felhasználásának minimális száma MINU, azaz

$$R'_i / (A_{ij} \text{ MINU}) \geq 1 \quad \text{ha } A_{ij} > 0.$$

Nyilvánvaló, hogy az előbb imsertetett elfogadási szintet meghatározó paramétereken /MAXTL, MINR-MAXR, MINU/ áll vagy bukik az eljárás eredményessége, hogyan tudjuk beállítani őket. Általánosan ezeket a paramétereket megadni nem lehet, mivel teljesen a konkrét üzemi problémától - az alapanyag és a rendelések szélességeinek a viszonyától, a tételszámok szórásától, stb. - függnnek. Intuición vagy korábbi eredmények alapján, felhasználva a manuális szabásminta készítés eredményeit, a kezdeti paramétereket el kell készíteni, egy szabásképet meg kell határozni, majd ezután a megmaradt rendeléseknél az /i/ és /ii/ becslések alapján esetleg a következő szabáskép elfogadási szintjét módosítani. Akkor is módosítani kell a paramétereket, ha nem találunk elfogadható szabásképet. Amennyiben a teljes megoldás elfogadhatatlan, akkor más induló paraméterekkel előlről el kell kezdeni az egészet, azaz iterációs lépésekkel jutunk el egy megfelelő megoldáshoz /HAESSLER elnevezése erre az eljárásra: "multiple-pass heuristic procedure - MPHP"/. Az "elfogadási szint"-nek eleget tevő szabásképet leszámplálással határozta meg, a rendeléseket tételszámuk szerint csökkenő sorrendbe állította és előlről haladva generálta az 1-3. feltételeknek eleget tevő szabásképet, elfogadva a legelső megfelelőt.

HASSLER az alkalmazásokkal kapcsolatban a következőket jegyezte meg [41] : a MAXTL általában 0,006 W és 0,03 W közötti érték, MAXR a kések számából adódó lehetséges csikok maximuma, MINR a homogén szabásképekben a rendelések átlagos számánál valamivel kisebb, míg MINU a rendelésállományhoz szükséges alaptekercsek számának 0,5 -től 0,9-ig terjedő része.



9. ábra. A szekvenciális heurisztikus eljárás blokkdiagramja.

HAESSLER próbafuttatásai az eljárás eredményességét igazolják, pl. a /15/ feladat esetén 16 %-kal csökkent a darabolás költsége a korábban alkalmazott kézi illesztgetésekhez képest. A /16/ feladat esetén egy rendelésállományra 14,6 inch /1 inch = 2,54 cm/ hulladék és 8 szabáskép lett, míg ugyanezrt a rendelésállományt GGM-el optimalizálva 3,9 inch lett a hulladék és 16 a szabásképek száma. A szabáskép váltásának és a hulladék költségének az aránya dönti el, melyik módszert érdemes használni. HAESSLER szerint a heurisztikus eljárás alkalmazása akkor előnyös, ha a rendelések átlagos mérete legalább hétszer fér fel az alapanyagra [49] .

A /15/ nem-lineáris darabolási feladat HAESSLER-féle megoldási módszerének javítására COVERDALE és WHARTON módosítást javasolt [17] . Alapgondolatuk az, hogy csak akkor fogadhatunk el és csak annyiszor alkalmazhatunk egy szabásképet, hogy a maradék rendelésállomány is "valószínűleg" jól kiszabható legyen. Véleményünk szerint az újratekerceselésnek és a felületvesztesség felső korlátja "lazításának" /új elfogadási paraméterek felállítása/ két alapvető oka van:

/i/ Amikor egy elfogadott szabáskép után a maradék rendelések aktuális tételszáma $/R'_1/$ olyan kicsi, hogy az őket tartalmazó későbbi szabásképet egyszer vagy kétszer lehet csak használni.

/ii/ A maradék szélességek olyan kicsik, hogy a kések kevés száma miatt újra kell tekercselni az egészet.

Az /i/ kiküszöbölésére az általuk készített felsorolási technikát /leszámlálás/ javasolták. Ennek lényege, hogy a szabásképek típusait definiálták aszerint, hányszor lehet alkalmazni az adott szabásképet és a kielégítetlen rendelésekből mennyi marad. Ezáltal csak olyan szabásképet fogadnak el, amely után még jól darabolható maradék keletkezik.

A /iii/ ellensúlyozására egy korlátozó paramétert készítettek. Legyen szabásképben szerepeltethető rendelések számának felső korlátja L , és mutassa X hányszor lehet használni ezt a szabásképet. Jelölje N az aktuális szabásképben a rendelések számát. Elfogadjuk ezt a szabásképet, ha az alábbi egyenlőtlenség fennáll:

$$W \left(\sum_i R_i - X N \right) / \left(\sum_i R_i W_i - X W \right) \leq L$$

azaz átrendezve,

$$N \geq \sum_i R_i / X - L \left(\sum_i R_i W_i - X W \right) / X W$$

A két módosítással HAESSLER eredményein jelentősen javítottak, legalábbis az általuk közölt futtatási eredmények alapján. A korábbi kézi illesztgetéshez képest 26-42 %-kal csökkent az összes költség /HAESSLER-nél ez átlagosan 16 %/ a konkrét feladattól, rendelésállománytól függően.



3.3. Módosítások és alkalmazások

Ebben az alponyban HAESSLER gyakorlati jellegű és JOHNSTON inkább elméleti alapokon álló eredményeit ismertetjük, vázoljuk röviden. JOHNSTON egy-dimenziós esetben az optimális szabáskepek számának alacsony szinten tartását a következőképpen próbálta elérni [62] : Az /1/ ILP feladathoz csatolta az alábbi nem-lineáris feltételt, amely az eltérő szabásminták számát korlátozza felülről:

$$\sum_{j=1}^n \delta_{/X_j/} \leq S_m$$

$$\text{ahol } \delta_{/X_j/} = \begin{cases} 1, & \text{ha } X_j > 0, \\ 0, & \text{egyébként.} \end{cases}$$

S_m az eltérő szabásminták számának felső korlátja.

A feltételek aggregálásával próbálta az így kibővített feladatot megoldani, azonban ez a próbálkozás gyakorlati eredmény nélküli volt.

Máshogy is meg lehet fogalmazni a minták váltásának redukálását [68] :

$$X_j \geq 0, X_j = \text{egész} \quad /j = 1, 2, \dots, n/$$

$$\sum_{j=1}^n A_{ij} X_j \geq N_i \quad /i = 1, 2, \dots, m/$$

$$\sum_{j=1}^n x_j \leq W_{\max}$$

$$\sum_{j=1}^n \sigma / x_j \longrightarrow \min!$$

ahol W_{\max} egy előre megadott korlát a veszteségre /egyféle L hosszú alapanyag van/. Ez a megközelítés szintén elméleti jellegű, gyakorlatilag csak heurisztikus eljárással lehet megoldani, amit vázolt is a "hard" feladat - amikor a rendelések hossza az alapanyag hosszához mérve elég nagy - ismertetése során [64] .

HAESSLER szekvenciális heurisztikus eljárására JOHNSTON indítványozott egy módosítást [65] , amelynek az a lényege, hogy a szabásképet hátizsákfeladat megoldásaként kapjuk meg - tehát a lehető legjobbat vesszük figyelembe -, míg HAESSLER a rendelések átrendezése után a legelső elfogadható szabásképnél megállt és nem kereste meg a legkisebb veszteségűt. Kisüzemeknél, ha kisszámú rendelést kevészer kell kiszabni és a szabásminták váltása is nehéz /általában 2 vagy 3 minta lehet csak/, alkalmazta az alábbi megfontolást. Alulról és felülről megbecsülte a szükséges alapanyag mennyiségét $/X' < X''/$ és kiszámolta, hogy a szabásképeket hányszor lehet alkalmazni. Pl. ha 2 eltérő szabáskép lehet csak, akkor meg kell határozni az összes párt, amely eleget tesz

$x' \cong x_1 + x_2 \cong x''$ -nek és $x_1 < x_2$, majd 0-1 feladattal meghatározta az optimális szabásképeket [63]. Kisüzemekben a gépek egyenletes leterhelésére pedig lineáris programozáson alapuló módszert adott [66].

Az egy-dimenziós darabolási feladat összes feltételét, típusait JOHNSTON egy összefoglalóban ismerteti [68], míg a számítógépes programokról, alkalmazásaikról szóló áttekintése [67]-ben található.

HAESSLER a tekercsek feldarabolásának problémáját átfogóan vizsgálja egyik cikkében [42]. Azt javasolja, hogy egy programrendszerben legyen lineáris programozáson /GGM/ és heurisztikus eljáráson alapuló rutin is, és a feltételek, elvárások pillanatnyi alakulásától függően kell az egyiket alkalmazni. Felvetette, hogy együtt a kettőt is lehet, egymás kiegészítéseként használni. A lineáris programozás optimális szabásképeinek döntő többségét el lehet fogadni - amelyek kis veszteségűek és viszonylag sokszor kell alkalmazni őket -, az utánuk maradó rendelésállományt viszont heurisztikusan kell feldarabolni és így a kerekítésekéből adódó hibák nagy részét el lehet kerülni, valamint a szabásminták számát is csökkenteni lehet.

HAESSLER ismerteti fémtekercsek olyan feldarabolását is, ahol az ő meghatározása szerint a feladat "1.5-dimenziós". Ugyanis egy rendelést a szélessége és a súlya határoz meg, a raktáron levő alaptekercsek szélessége állandó, de viszont

mindegyiknek más-és-más a súlya. Ez a feladat valóban bonyolultabb az egy-dimenziósnál [44]. Itt is a hulladék minimalizálása és a szabásminták alacsony száma a cél, amihez szintén a heurisztikus szekvenciális eljárást használja.

Eddig egy szabásgépről /tekerceselőről/ volt szó, de a nem-lineáris modell több-gépes darabolás esetén sem változik lényegesen, csak a gépek egyenletes leterhelését is figyelembe kell venni, amint az HAESSLER vázolja [48]-ban.

Acéltekercesek felhasításakor [46], de műanyag lemezek, filmek darabolásakor is [43], [45], a gyártógépből kikerülő anyagon egy szabályozható elővágást végeznek, majd az így megkapott tekercesekből szabják ki a rendeléseket - ezt két-üteműnek nevezte. Előnye, hogy az elővágással a visszatekerceselés megszüntethető és így lényegesen gyorsabb lesz a darabolás.

Egy-dimenziós esetekben is felvetődik a rendelések folyamatos leszabásának kérdése, amit HAESSLER és TALBOT egy nagyméretű bináris mátrix segítségével próbáltak megoldani [47]. Sajnos, elgondolásuk két-dimenziós esetekre nehezen vihető át.

Az előbbieken olyan egy-dimenziós darabolási feladatokat ismertettünk, amelyeknek célfüggvénye nem-lineáris. A vázolt heurisztikus szekvenciális eljárás előnye, hogy kis számító-

gépen is gyorsan készíti el a megfelelő szabásképeket, a raktárkészletet könnyü figyelembe venni és a felhasználók közvetlen kapcsolatban lehetnek a számítógéppel - lényegében a kézi programozást szimulálja az eljárás. Hátránya, hogy az egyedi /időben is sűrűn változó/ feltételekhez nagyon kötődik, a paraméterek beállítása részben szubjektív becslésen nyugszik.

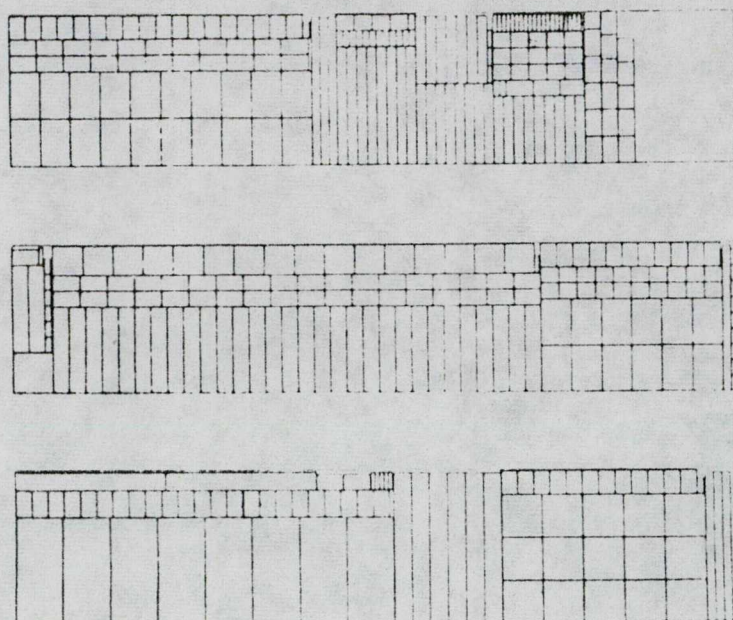
A következő fejezetben egy két-dimenziós feladatot és a megoldására javasolt heurisztikus módszert ismertetjük.

4. Nagyméretű alaptáblák heurisztikus feldarabolása

Ebben a fejezetben bemutatunk egy olyan heurisztikus módszert, amely sokféle, de kis tételszámu és az alaptáblához viszonyítva kisméretű rendelkezések kiszabására előnyösen alkalmazható. Általában 2-3 szabásképet kell elkészíteni, és mindegyiket csak egyszer kell alkalmazni.

Az ilyen jellegű feladatot meg lehetne oldani a GGM-el is, de a hátizsákfeladat hely- és időigénye ennél a problémánál nagyon nagy lenne, másrészt az az egészértékűségekre való kerekítés a megoldást teljesen eltorzíthatja. A módszer téglalapból-téglalapot-ortogónálisan esetre készült, de később kitérünk más jellegű alkalmazásokra is. Több-ütemű quillotine-vágásokból állnak a szabásképek, a rendelkezések balra vannak tömörítve, így hasznos hulladékot is kapunk. A 10. ábrán látható három szabásminta, amelyet ALBANO és ORSINI közölt [7] egy adott feladat megoldásaként. Mindegyik szabásképet csak egyszer kell le-
szabni.

A módszert ADAMOWICZ és ALBANO [2], valamint ALBANO és ORSINI [7] cikke alapján ismertetjük. Munkáikban felhasználták HAIMS és FREEMAN eredményeit [50], módszer kevésbé kidolgozott változata pedig [1] -ben található.



10. ábra. A nagyméretű alaptáblák feldarabolására jellemző szabásképek [7] .

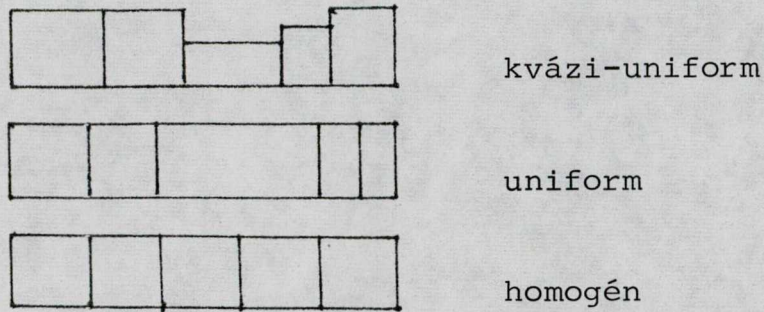
Alapgondolatuk a következő megfigyelésen, észrevételen alapszik. A darabolást programozók, akik kézi illesztgetéssel készítik el a szabásmintákat, ilyen nagyméretű alaptábla esetén először a rendelésekből csikokat, a csikokból blokkokat hoznak létre és a blokkok egymás mellé illesztésével kapják meg a szabásmintákat. Az eljárásuk ezt a gondolatot fordítva követi, először az alaptáblából ún. "al-táblákat" hoznak létre, majd ezeket az al-táblákat csikokkal kitöltik. Az alapvető

és a legnehezebb lépés az alaptáblák felosztása al-táblákra - ehhez kellene heurisztikus megfontolások -, mivel a továbbiakban egy al-tábla kitöltése egzakt módon, hátizsákfeladat megoldásaként megkapható. Az al-táblák kitöltését két-ütemű feladatként kezelik, de általános quillotine-vágást szolgáltató eljárást is lehet használni /pl. HERZ [53] vagy CHRISTOFIDES és WHITLOCK [12] algoritmusát/. A továbbiakban egy al-tábla kitöltését, a csikok elkészítését ismertetjük, majd vázoljuk a heurisztikus lépések sorozatát.

4.1. Az al-táblák feltöltése

Legyen l_i, w_i, n_i a rendelt téglalap hossza, szélessége és tételszáma $/i = 1, 2, \dots, N/$. A rendeléseket és méreteiket rendezzük át, hogy teljesüljön $l_i \geq w_i /i \leq N/$ és $l_i \geq l_{i+1} /i < N/$. Az al-tábla hossza legyen L , a szélessége pedig W .

Az első ütemben csikokat generálunk, amelyekből azután programozással választjuk ki a megfelelőket. A csikok elkészítésekor több alsó elfogadási-korlátot használunk, amelyekkel a lehetséges csikok számát csökkenthetjük. Legyen A a csikban számításba vehető legnagyobb felületű rendelés felülete, ekkor $T_n * A$ a csikban szereplő rendelések felületére, $T_l * L$ és $T_w * W$ a csik hosszára és szélességére ad alsó korlátot. A csikok elkészítése a következő lépésekből áll /a lehetséges csikokat, megnevezésüket a 11. ábra mutatja/:



11. ábra. A lehetséges csikok felépítése.

1. lépés: A leghosszabb elfogadható homogén csik elkészítése az i -edik rendeléshez $\langle n_i \rangle > 0$ és a hosszának $\langle L_i \rangle$ a kiszámítása:

$$u_1 = \max \quad \langle \ell_i * m_i \rangle$$

$$u_2 = \max \quad \langle w_i * m'_i \rangle$$

$$L_i = \max \langle u_1, u_2 \rangle$$

ahol $m_i, m'_i \leq n_i$ egészek

$$\ell_i * m_i \leq L \text{ és } w_i \leq W,$$

$$w_i * m'_i \leq L, \quad \ell_i \leq W \text{ és } \ell_i * w_i \geq T_n * A.$$

2. lépés: Ha $L_i \geq T_e * L$ és $w_i \geq T_w * W$ akkor menjünk a 4. lépésre. Azaz elég "jó" homogén csikot fogadunk el.

3. lépés: A homogén csik hosszát próbáljuk növelni más rendelések segítségével. A w_k szélességű rendelést csatoljuk a csikhoz $\langle n_k \rangle > 0, w_k \leq w_i, k \neq i$, ha $w_k \geq T_h * w_i$, ahol T_h egy elfogadási alsó korlát.

Ebben a lépésben az uniform és a kvázi-uniform csikokat készítjük el. Ha L_i nem növelhető, akkor elhagyjuk ezt a homogén csikot és megyünk az 1. lépésre.

4. lépés: A jelölt csikok közé felvesszük ezt a csikot. Kiszámoljuk a csik alkalmazhatóságának maximális számát, b_i -t és a csikban szereplő rendelések összfelületét, s_i -t. Ha figyelembe nem vett rendelés van még, akkor visszamegyünk az 1. lépésre.

A fentiekben elkészített csikok összeillesztését - legyen a számuk: r -, a hátizsákfeladatot dinamikus programozással oldjuk meg az alábbi függvény kiszámításával:

$$G_i(x) = \max [G_i(x - k * w_i + 1) + k * s_i]$$

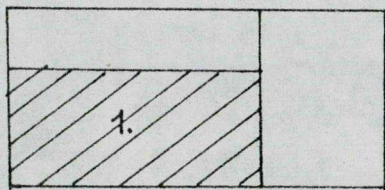
ahol $0 \leq x \leq W$ és

$$0 \leq k \leq \min [b_i, \lfloor W/w_{i+1} \rfloor] \quad / \quad /i = 1, 2, \dots, r/$$

Legyen $G_0(x) = 0$ minden x -re. A $G_r(W)$ kiszámolásával az $L * (r + 3)$ rekesz. Ezt a dinamikus programozáson alapuló eljárást és számítógépes háttérét ALBANO és ORSINI egyik munkájában részletesen tárgyalja 6 .

4.2. Az alaptábla felosztása al-táblákra

Az alaptábla felosztására készítendő stratégia nagyban függ a konkrét megoldandó problémák jellegétől, az alaptábláktól és a rendelkezésektől. A következőkben röviden vázoljuk azt az elgondolást, amely a legnagyobb területű rendelést próbálja meg elhelyezni, ahhoz készíti el az al-táblákat - természetesen ez a lehetséges stratégiák közül csak egy -. Készítsünk a legnagyobb területű rendelésből homogén csikot és a csik hosszának megfelelően vágjuk ketté az alaptáblát. Ezt követően a homogén csikot tartalmazó al-táblát a 4.1.-ben vázoltaknak megfelelően töltjük ki. Ha ez a kitöltés nem fogadható el, akkor vagy próbáljuk az al-táblát a legnagyobb területű rendelés kisebb homogén csikja alapján elkészíteni, vagy a homogén csikok egymás mellé illesztésével kapott téglalap mentén vízszintesen elvágjuk az al-táblát /a 12. ábrán az 1. jelöli az így kapott al-táblát/, stb. Akkor fogadunk el egy al-tábla feltöltést, ha a hasznos felület aránya nagyobb /azaz a veszteség kisebb/ egy előre megadott T_u korlátnál.

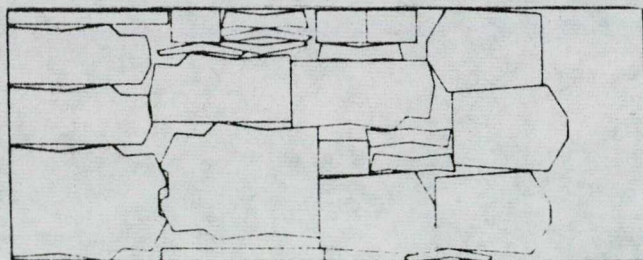


12. ábra. Az al-tábla generálása.

A megoldás "jósága" és gyorsasága nagyban függ a csikok szelektálását, elfogadását szabályozó alsó korlátoktól. A próbafuttatások alapján az alábbi értékeket alkalmazták:

$$T_n = 5 - 15 \%, T = 90 \%, T_w = 10 \%, T_h = 90 \% \text{ és } T_u = 90 \%.$$

A téglalapról-téglalapra adott eljárás alap-gondolatát adaptálták téglalapról szabálytalan idomok kiszabására is. Szintén al-táblákat töltöttek fel /az al-táblák téglalapok/, majd ezek alapján készítették el a szabásmintákat. Ez a megközelítés található az [1] , [3] és a [4] munkákban. A szabálytalan idomok kiszabásának más megközelítését alkalmazta ALBANO és SAPUPPO [5] , amikor NILSSON eredményei [92] alapján a szabásminták elkészítésére egy fabejáráson alapuló heurisztikus algoritmust készítettek /az [5] -ből vett 13. ábra egy jellemző szabásmintát tartalmaz/. Az ilyen jellegű feladat elsősorban a textil- és bőriparban vetődik fel, de jelentkezhet pl. fémlemezek darabolásakor, sajtolásakor is.



13. ábra. Szabálytalan idomok kiszabására egy jellemző szabásminta [5] .

A nagyméretű alaptáblák feldarabolásánál ez a heurisztikus eljárás elfogadható gépidő alatt a darabolás gyakorlati feltételeihez - amire jelen esetben a GGM nem alkalmazható - jól illeszkedő megoldást szolgáltat. A heurisztikát szabályozó paraméterek megfelelő beállítása és az alaptáblák al-táblákra bontásának stratégiája dönti el adott esetben az eljárás "jóságát",



5. A síküveg szabásának optimalizálása az Orosházi Üveggyárban

A fejezetben az Orosházi Üveggyár részére készített /a [78] diplomamunkában és a [79] publikációban részletesen leírt/ számítógépes eljárást ismertetünk, amellyel a síküveg-szabásánál keletkező felületveszteséget minimalizálni lehet. Természetesen a darabolási folyamat optimalizálásának ez csak egy bizonyos szempontból való megközelítése, más szempontok /szabáshoz szükséges idő, raktározás, anyagmozgatás, stb./ esetleg mérőben más jellegű feladatok megoldását igénylik. A modellezés során a gyakorlati szempontokat részesítettük előnyben.

A síküveg hagyományosnak tekinthető eddig alkalmazott szabásánál gyári szakemberek becslése szerint kb. 33-35 % hulladék keletkezik, ezzel szemben a számítógép által adott szabástervek vesztesége 20-22 %. Vagyis a számítógépes eljárás a hagyományos szabási módszerhez képest kb. 13 %-kal csökkentheti a hulladékot, vagy másképpen megfogalmazva, ugyanannyi alapanyag feldarabolása esetén kb. 20 %-kal nő a hasznos, eladható üvegfelület. A módszer szorosán illeszkedik a jelenlegi technológiai folyamathoz, műszaki változtatások, beruházások nélkül alkalmazható bizonyos átszervezések végrehajtása után.

A rendelkezések sürgősségi jellegük folytán két csoportra oszthatók: a napi sürgős és az előre ismert, hosszabb határidejű rendelkezésekre. A napi sürgős rendelkezések kis tételszámuk és szinte azonnal hozzá kell kezdeni a leszabáshoz, itt az optimalizálás nagy nehézségek leküzdése árán is csak viszonylag gyengébb eredményekkel kecsegtet. Az üvegyár jelenleg nem rendelkezik számítógéppel, így az optimális szabásterv elkészítése és a tényleges leszabás csak különböző helyeken /pl. számítóközpontban és a szabász-üzemben/ és időben egymáshoz képest esetleg több napos eltéréssel végezhető el, amiből következik, hogy csak a hosszabb határidejű rendelkezések darabolásának optimalizálása jöhet szóba. Ezek a rendelkezések az összes mennyiségnek a nagyobb hányadát alkotják és általában elég nagy a tételszámuk /legalább 50/.

A gyárban jelenleg kétféle hagyományos módon darabolják fel az alaptáblákat. Kézi szabásnál a vágófejeket és a téglalapok elhelyezkedését a szabással foglalkozó munkás kézzel állítja be, így a felületveszteség nagyon függ a szakértelmétől és a tapasztalatától. A másik hagyományos szabás a sorozatvágás, ahol egy automata vágógép gyorsan és pontosan darabolja, pneumatikusan mozgatja az üvegtáblát, de itt nincs lehetőség a hulladék lényeges csökkentésére, mivel egy alaptáblából csak egyféle téglalapot tud kivágni. A szabás megkezdése előtt kiszámítják az adott rendeléshez legmegfelelőbb alaptáblát és a raktárból bekészítik. Ezeknél a hagyományos szabási módszereknél a felületveszteség kb. 33-35 %. A sorozatvágás nagy

vesztesége elsősorban abból származik, hogy nem teszi lehetővé különböző méretű téglalapok egyidejű leszabását.

A siküveg szabásánál alkalmazott szabásgépek két-ütemű quillotine-vágást tesznek lehetővé.

Megoldásként a legegyszerűbbnek tűnő eljárást választottuk: az összes szóba jöhető szabásképet elkészítjük és egészértékű lineáris programozással /ILP/ választjuk ki közülük az optimális szabásképeket. Természetesen az összes szabáskép elkészítése és a nagyméretű ILP feladat kezelése okozza az igazi nehézségeket. Az összes szabásképet mindig a konkrét rendelésállomány ismeretében egy gyors kombinatorikus algoritmussal készítjük el, az ILP feladatot pedig programkönyvtárban levő programcsomag segítségével oldjuk meg.

Miért döntöttünk úgy, hogy ezzel a módszerrel keressük meg az optimális szabásmintákat? Az alábbi szempontokat vetjük figyelembe:

- az üvegszabás feltételei annyira speciálisak /a két-ütemű quillotine-vágás/ - a két-dimenziós feladat itt lényegében két egy-dimenziós feladatra redukálódott -, hogy a feltételeket jól kihasználó, egyszerű kombinatorikus eljárás készíthető a szabásképek előállítására,

- a szabásképek száma bizonyos redukciók és a félkésztermékek méreteinek kicsiny száma miatt nem nagy /1000-1500-nál nem több/,
- a félkésztermék-raktár kapacitásának figyelembe vétele egyszerűen nagy, az ILP bővül néhány feltétellel,
- fel lehet használni a programkönyvtárban levő optimalizáló programcsomagokat - nem kell sok munkát igénylő és bonyolult optimalizáló programokat megírni és "belőni", mivel nem-lineáris feltételek is vannak -, ezért a szabást optimalizáló rendszer rövid idő alatt és főleg olcsón elkészíthető,
- bármely másik, legalább "közepes" nagyságu számítógépre az eljárás könnyedén "áttelepíthető", mivel ILP /esetleg LP/ feladatra általában minden programkönyvtárban van megoldó rutin.

5.1. Az üvegszabási probléma leírása

A következőkben az üvegtáblák két-ütemű quillotine-vágásának speciális gyári feltételeit ismertetjük.

A rendelések paraméterei a következők: vastagság, minőség, szélesség, hosszúság és tételszám. A félkésztermékek paraméterei ugyanezek. Természetesen egy rendelést csak az ugyanolyan vastagságu és minőségü alaptáblákból lehet kiszabni.

Tehát a minőség és a vastagság a rendelések halmazát olyan diszjunkt részhalmazokra bontja, amelyeket egymástól függetlenül kell szabni. Következésképpen ezekre a részhalmazokra külön-külön elkészített optimális szabástervek együtt alkotják az egész rendelésállomány optimális szabástervét.

A továbbiakban egy csoporton belül vizsgáljuk a problémát.

Legyen a csoport rendeléseinek halmaza: R ,

$$R = \left\{ r_i \mid r_i (w_i, \ell_i, N_i) , i = 1, 2, \dots, m \right\} ,$$

ahol: m a csoport rendeléseinek a száma,

w a szélessége,

ℓ a hosszúsága,

N a tételszáma a rendelésnek.

A megfelelő félkésztermékek halmaza: F ,

$$F = \left\{ F_j \mid F_j (W_j, L_j, Q_j) , j = 1, 2, \dots, p \right\} ,$$

ahol: p az alaptábla-féleségek száma,

W a szélességét,

L a hosszúságát,

Q a darabszámát jelöli a félkészterméknek.

Adott vastagsánál az alaptáblák szélessége a műszaki feltételek következtében állandó, ezért

$$W = W_j \quad (j = 1, 2, \dots, p) .$$

Az optimalizálást a két-ütemű quillotine-vágásra készítettük el, amelynek a modellezés szempontjából fontos üvegyári jellemzői az alábbiak:

- /i/ az alaptábla egyik oldalával párhuzamosan csikokat vágunk /elsőütem/, majd a csikokat egyenként daraboljuk fel a kívánt méretre /második ütem/,
- /ii/ A csikok száma legfeljebb S_1 , egy csikon belül a téglalapok száma pedig legfeljebb S_2 lehet /erre azért van szükség, mivel a vágófejek miatt egy asztalon legfeljebb ennyi vágást lehet egyszerre elvégezni/,
- /iii/ az eltérő méretű téglalapok száma egy szabásképben legfeljebb T /erre a feltételre azért volt szükség, mert az üzemben a vágóasztalok mellett csak meghatározott számú rendelés tárolható és csomagolható egyszerre/.

Az üvegszabás feladata: Keresendő olyan eljárás, amely a konkrét üzemi feltételeket figyelembe véve, az előre megadott rendelésekre - lehetőleg a félkésztermékek raktárkészletét is felhasználva - olyan szabástervet szolgáltat, amelynek maradóktalan teljesítése esetén a felületveszeség minimális lesz.

5.2. A leszabás matematikai modellezése

A feladatot - mint azt már említettük korábban - két lépésben oldjuk meg:

1. fázis: Az összes szóba jöhető, a feltételeknek eleget tevő szabáskép elkészítése,
2. fázis: a szabásképekből az optimális szabásminták meghatározása.

A szabás első lépésében a csikok levágása az alaptábla szélességével párhuzamosan történik.

A következőkben megadjuk az összes szabásképet és jellemzőiket. Először az összes lehetséges csikot készítjük el, majd a csikok egymás mellé illesztésével az összes lehetséges szabásképet határozzuk meg.

5.2.1. Az összes szabáskép meghatározása

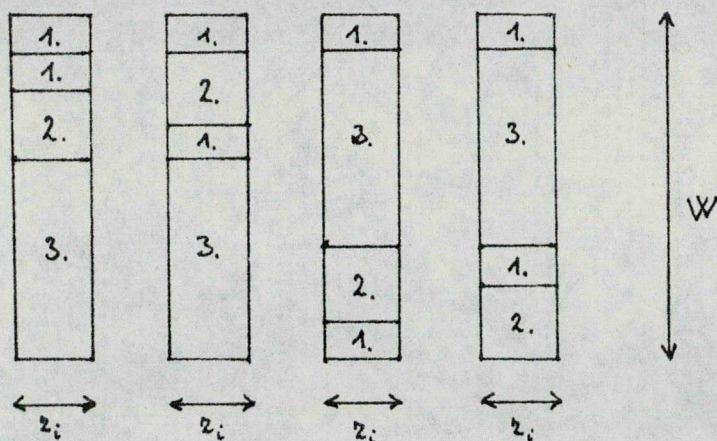
A csikok szélessége a $w_1, \ell_1, w_2, \ell_2, \dots, w_m, \ell_m$ méretek közül kerül ki, hiszen egy téglalap legfejlebb kétféle helyzetben, a w és az ℓ méretének megfelelő szélességű csikban szabható le. Az előbb felsorolt méretek között lehet megegyező is, ezért z_1, z_2, \dots, z_d / $d \leq 2m$, $z_i < z_j$, ha $i < j$ / jelölje a csik-szélességként szóba jöhető szélesség- és hossz méretek olyan együttes felsorolását, ahol már minden szám különböző.

A z_i -hez / $i=1, 2, \dots, d$ / hozzárendelünk több méretet, mégpedig azoknak a téglalapoknak a másik méretét, amelyekben a z_i szélességként vagy hosszúságként szerepel. Azaz z_i -hez

hozzárendeljük z^i -t:

$$z^i = \{z_i^1, z_i^2, \dots, z_i^M\} \quad (1 \leq M \leq m, z_i^k < z_i^j \text{ ha } k < j)$$

Erre azért volt szükség, mivel egy csikban csak olyan rendelések szerepelhetnek, amelyeknek egyik mérete megegyezik a csik szélességével, ezért a z_i szélességű csikban levő téglalapok egyik mérete z_i , a másik mérete pedig a z^i elemei közül kerül ki. A csikon belül a téglalapok sorrendje tetszőleges, ezért pl. a 14. ábra csikjai a leszabás optimalizálása szempontjából azonosnak tekinthetők.



14. ábra. A csikok azonosak a leszabás optimalizálása szempontjából

Mivel a csikon belül a téglalapok sorrendje tetszőleges, ezért a csikot egyértelműen meghatározza az, hogy melyik megrendelésből hány szerepel benne. Következésképpen minden csikot jellemezhetünk egy vektorral, az ún. "csikösszeállítási-vektorral", egyszerűbben a csik-vektorral.

Legyen a csik-vektor $C_j^i = (c_{1j}^i, c_{2j}^i, \dots, c_{nj}^i)^*$

ahol c_{kj}^i mutatja, hogy a k-adik rendelésből hány van a z_i szélességű j-edik csikban.

A következő hat feltételnek kell eleget tenni:

$$/17/ \quad c_{kj}^i \geq 0 \quad /k = 1, 2, \dots, m/;$$

$$/18/ \quad \text{ha } c_{kj}^i > 0, \text{ akkor } z_i = w_k \text{ vagy } z_i = l_k$$

$$/19/ \quad \sum_{k=1}^m c_{kj}^i \leq S_2$$

- legfeljebb ennyi téglalap lehet egy csikban;

$$/20/ \quad \sum_{k=1}^m \text{sign } c_{kj}^i \leq T$$

- legfeljebb ennyiféle rendelés lehet egy csikban.

Legyenek $c_{k_1j}^i, c_{k_2j}^i, \dots, c_{k_\alpha j}^i$ a C_j^i nemzérő komponensei és $b_{k_s}^i$ az r_{k_s} rendelés z_i -től eltérő mérete /négyzet alakú

téglalagnál z_i -vel egyenlő/ $/s = 1, 2, \dots, \alpha/$, akkor

$$/21/ \quad \sum_{s=1}^{\alpha} b_{k_s}^i c_{k_s j}^i \leq W$$

- a csikban levő téglalapok szélességének az összege nem haladhatja meg az alaptábla szélességét;

$$/22/ \quad \text{ha } W - \sum_{s=1}^{\alpha} b_{k_s}^i c_{k_s j}^i \geq z_i^1$$

$$\text{akkor } \sum_{k=1}^m \text{sign } c_{kj}^i = T \text{ vagy } \sum_{k=1}^m c_{kj}^i = S_2$$

- a hulladékból több megrendelést már nem tudunk levágni.

Képezzük z_i -hez az összes lehetséges csik-vektort, ezáltal meghatározzuk az összes z_i szélességű csikot.

A továbbiakban a C_j^i csik veszteségén a

$$v_j^i = z_i \cdot W - \sum_{k=1}^m c_{kj}^i \cdot w_k \cdot \ell_k$$

mennyiségét értjük.

Tegyük fel, hogy a z_1, z_2, \dots, z_d szélességekhöz az összes csik-vektort már meghatároztuk. Ki kell szűrni közülük azokat, amelyek nem szerepelhetnek az optimális szabástervben.

Egymás után a következő redukciós szempontokat vettük figyelembe:

/i/ Ha egy rendeléshez két homogén csik tartozik /homogén egy csik, amikor csak egyféle téglalap van a csikban, azaz $\sum_k \text{sign } c_{kj}^i = 1$ /, pl. r_k -hoz a w_k és az ℓ_k szélességű, és

$$\left[\frac{W}{w_k} \right] \leq \left[\frac{\ell_k}{w_k} \right] \left[\frac{W}{\ell_k} \right],$$

akkor az ℓ_k szélességű csikot elhagyjuk / [] az egész-rész jele/. Ugyanis, ha az optimális szabásminták valamelyikében szerepelne ez az ℓ_k szélességű csik, akkor a w_k szélességűt a helyére rakva nem növelnénk a veszteséget.

/ii/ Minden csik esetén megnézzük, vannak-e olyan csikok, amelyekkel ugy helyettesithető, hogy ugyanazok a megrendelések szerepelnek az új csikokban is, de a veszteség már kisebb lesz. Ez az előbbi /i/ redukciónak az általánosítása /azért vettük külön, mivel az /i/ az egyszerűsége miatt a számítógépes programban is külön szerepel/. Egy ilyen helyettesítés több különböző csikból állhat, de csak olyanokból, amelyeknek a vizsgált csik szélességénél nem nagyobb a szélességük és amelyekben a vizsgált csik rendelésein kívül más rendelés nem szerepel. A vizsgált csikot elhagyjuk, ha találunk a veszteségnél nem nagyobb veszteségű helyettesítést. /Feltételezzük, hogy a vizsgált csik olyan sokszor szerepel a szabás-tervben, hogy mindegyik behelyettesítendő csikot legalább egyszer lehet alkalmazni/.



Például: legyen a helyettesítendő csik: $c_1^i \left(\sum_k \text{sign } c_{k1}^i = 2 \right.$
 és $c_{11}^i > 0, c_{21}^i > 0$), tegyük fel, hogy az optimális szabáster-
 ven B_1 -szer szerepel és a vesztesége: v_1 . Ekkor a téglalapokból
 $B_1 c_{11}^i$ és $B_1 c_{21}^i$ darabot kell levágni.

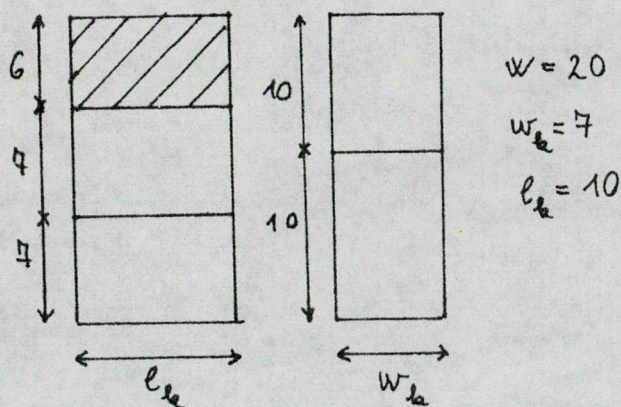
Legyen c_2^i egy behelyettesítő csik $\left(\sum_k \text{sign } c_{k2}^i = 2 \right.$ és $c_{12}^i > 0,$
 $c_{22}^i > 0$), amelynek a vesztesége: v_2 .

$$\text{Ekkor } B_2 = \min_{k=1,2} \left\{ \frac{B_1 c_{k1}^i}{c_{k2}^i} \right\}$$

-szor lehet ezzel a csikkal behelyettesíteni az eredetit.

/Bevezettük a $\lceil \rceil$ jelölést, amely bármely E valós számra

$$\lceil E \rceil = \begin{cases} \lceil E \rceil & \text{ha } E - \lceil E \rceil = 0, \\ \lceil E \rceil + 1, & \text{egyébként.} \end{cases}$$



15. ábra. Numerikus példa az l_k szélességű homogén csik elhagyására

Tegyük fel, hogy $B_2 \approx \frac{B_1 \cdot c_{11}^i}{c_{12}^i}$

ekkor r_2 -ből maradt $B_1 \cdot c_{21}^i - \frac{B_1 \cdot c_{11}^i}{c_{12}^i} c_{22}^i$

darab. Ezt a mennyiséget homogén csikból vágjuk ki, amely $\sum_k \text{sign } c_{k3}^i = 1$ és $c_{23}^i > 0$, a veszteség: v_3 . A homogén csikot B_3 -szor kell venni,

$$B_3 \approx \frac{1}{c_{23}^i} \left(B_1 c_{21}^i - \frac{B_1 \cdot c_{11}^i}{c_{12}^i} c_{22}^i \right)$$

Az eredeti csikot elhagyjuk, ha

$$B_1 v_1^i \geq B_2 v_2^i + B_3 v_3^i,$$

azaz behelyettesítve és B_1 -el egyszerűsítve:

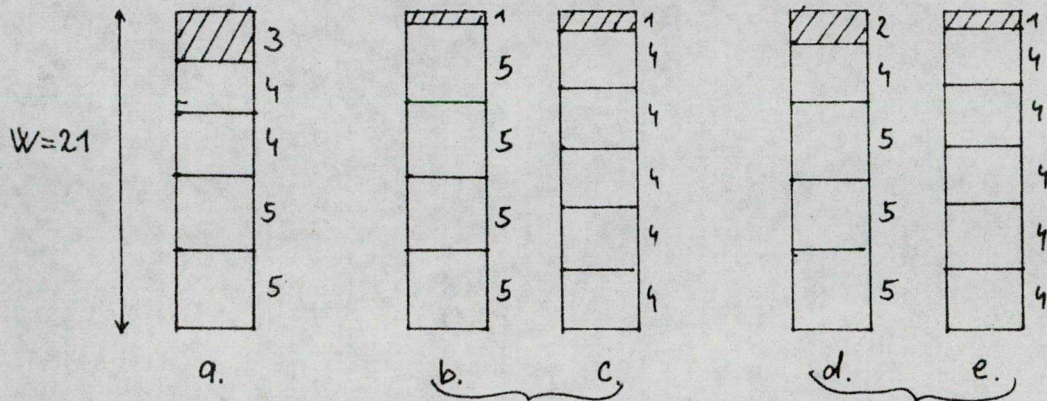
$$/23/ \quad v_1^i \geq \frac{c_{11}^i}{c_{12}^i} v_2^i + \left(\frac{c_{21}^i}{c_{23}^i} - \frac{c_{11}^i c_{22}^i}{c_{12}^i c_{23}^i} \right) v_3^i$$

Ebben a képletben már minden mennyiség ismert. Természetesen c_2^i homogén csik is lehet. A $T \geq \sum_k \text{sign } c_{kj}^i > 2$ esetek is hasonlóan vizsgálhatók.

Megjegyzés: az egészértékűséget a képletek megadásánál figyelmen kívül hagyjuk.

Példa: Ha a 16. ábrán levő a/ csik 10-szer szerepel a szabásokban, akkor a b/ csikot 5-szor, a c/ csikot pedig 4-szer kell alkalmaznunk ahhoz, hogy ugyanazokat a megren-

deléseket kisebb veszteséggel állítsuk elő. Hasonlóképpen, ha az a/ csik 15-ször szerepel a szabásképekben, akkor 10 db d/ csik és 4 db c/ csik kisebb veszteséggel pótolja.



16. ábra. Numerikus példa a /ii/ csik-helyettesítésre

Az összes csik-vektor elkészítése és az előbb leirt redukciós lépések végrehajtása után megkaptuk a szabásrajzok képzéséhez szükséges összes csikot és jellemzőiket. A következőkben a szabásképek előállítását ismertetjük.

Állítsuk sorba a csik-vektorokat:

$$/24/ \quad c_1^1, c_2^1, \dots, c_{b_1}^1, c_1^2, c_2^2, \dots, c_{b_2}^2, \dots, c_1^d, c_2^d, \dots, c_{b_d}^d$$

ahol b_i a redukció után megmaradt z_i szélességű csikok száma.

Az összes szabásképet a csikok egymás mellé illesztésével készítjük el. Egy szabásképen belül a téglalapok sorrendje, elhelyezkedése az optimalizálás szempontjából tetszőleges,

ezért a szabásképet egyértelműen meghatározza, hogy melyik téglalapból hány szerepel benne. Ebből következik, hogy az ún. "szabáskép-összeállítási-vektorral", /a korábban beneveztett/ szabás-vektorral a szabásrajzot egyértelműen jellemezhetjük.

A csik-vektor /24/ felsorolásából kell kiválasztani azokat, amelyek egymás mellé illesztésével szabásképet akarunk meghatározni. Nézzünk egy kiválasztást, amelyben a C_{j1}^{d1} csik-vektor q_{j1} -szer, ..., a $C_{j\alpha}^d$ csik-vektor $q_{j\alpha}$ -szor szerepel.

A kiválasztás által meghatározott szabás-vektor /legyen ez a j-edik/ az

$$a_j = (a_{1j}, a_{2j}, \dots, a_{mj})^* = \sum_{k=1}^{\alpha} q_{jk} C_{jk}^{dk}$$

összegvektor, ahol a_{ij} azt mutatja, hogy r_i -t hányszor kell levágni ebben a szabásképben.

A szabás-vektoroknak a következő négy feltételt kell kielégíteniük:

/25/
$$\sum_{k=1}^{\alpha} q_{jk} \leq S_1$$

- legfeljebb S_1 csikből állhat;

/26/
$$\sum_{k=1}^m \text{sign } a_{kj} \leq T$$

- legfeljebb ennyiféle rendelés lehet egy szabásképben;

/27/
$$\sum_{k=1} q_{jk} z_{dk} \leq L_p$$

- a csik-szélességek összessége a legnagyobb alaptábla-hossznál kisebb.

A a_j -hez hozzárendelünk egy alaptábla-méretet, L_h^j -t:

$$L_h^j = \min \left\{ L_h \mid \sum_{j=1}^{\alpha} q_{j_k} z_{d_k} \leq L_h, h = 1, 2, \dots, p \right\}.$$

/28/ ha $L_h^j - \sum_{k=1}^{\alpha} q_{j_k} z_{d_k} \geq z_1$

akkor $\sum_{k=1}^{\alpha} q_{j_k} = S_1$ vagy $\sum_{k=1}^m \text{sign} a_{j_k} = T$

- vagyis ha egy újabb csikot lehetne illeszteni a meglevők mellé, akkor ezt már csak a /25/ vagy a /26/ feltételek megsértésével lehetne.

Az a_j -hez hozzárendelünk egy veszteséget /hulladékot/, amely

$$H_j = W L - \sum_{k=1}^m a_{k_j} \cdot w_k \cdot l_k$$

A lineáris programozási feladat felírásához a szabásképek előbb meghatározott paramétereire van szükség. A szabásképek várhatóan nagy száma miatt itt is egyszerű redukciókat hajtunk végre /a csik-redukcióhoz hasonlóan/ elhagyva azokat a szabásképeket, amelyek nem szerepelhetnek az optimális szabásminták között.

/i/ A j-edik homogén szabásképet elhagyjuk, akkor

$$\left(\sum_s \text{sign} a_{sj} = 1 \text{ és } a_{kj} > 0, \text{ ha van olyan } a_i \text{ homogén szabáskép } \sum_s a_{si} = 1 \text{ és } a_{ki} > 0, \text{ hogy } L_h^j \cdot a_{ki} > L_h^i \cdot a_{kj} \right).$$

Egyenlőség esetén csak az egyiket tartjuk meg.

/ii/ A megmaradó szabásképeknél megnézzük, van-e olyan helyettesítés, amelyet a szabáskép helyére téve a veszteség nem nő, ha van ilyen, akkor ezt a vizsgált szabásképet elhagyjuk. A helyettesítés több különböző szabásképből állhat, de csak olyanokból, amelyekben a helyettesítendő szabásrajz megrendeléseinek kivül más rendelés nem szerepel /az előbbi redukciós feltétel ennek speciális esete/.

Például: Az a_1 szabásképet elhagyjuk / $\sum_j \text{sign } a_{j1} = 2$ és $a_{11} > 0, a_{21} > 0, a_{22} > 0$ a vesztesége: H_2 /és a_3 homogén szabáskép / $\sum_j \text{sign } a_{j3} = 1$ és $a_{23} > 0$ a vesztesége: H_3 /, hogy

$$H_1 \geq \frac{a_{11}}{a_{12}} \cdot H_2 + \left(\frac{a_{21}}{a_{23}} - \frac{a_{11} a_{22}}{a_{12} a_{23}} \right) H_3$$

A $T \geq \sum_j \text{sign } a_{ji} > 2$ eset behelyettesítésének képlete is hasonlóan adódik. Az egészértékűségtől itt is eltekintettünk.

A redukciók után megmaradt szabás-vektorok száma legyen: n , a sorrendjük pedig rögzített:

$$a_1, a_2, \dots, a_n$$

ahol H_j és L_h^j a megfelelő veszteség, ill. alaptáblahossz / $j = 1, 2, \dots, n$ /.

5.2.2. Az egészértékű lineáris programozási feladat

A fent leírt módon megkaptuk az összes szóba jöhető szabásképet. Ezután az alábbi ILP megoldásaként megkapjuk az optimá-

lis szabásmintákat. Az ILP a következőképpen írható fel /a /7/ feladat analogonja/:

$$x_j \geq 0, x_j = \text{egész} \quad /j = 1, 2, \dots, n/$$

$$\sum_{j=1}^m a_{ij} x_j \geq N_i \quad /i = 1, 2, \dots, m/$$

/29/

$$\sum_{j=1}^m \sigma_{kj} x_j \leq Q_k \quad /k = 1, 2, \dots, p/$$

$$\sum_{j=1}^m L_h^j x_j \rightarrow \min!$$

ahol n a szabásképek száma

m a rendelések száma,

a_{ij} azt mutatja, hogy az i -edik rendelés a j -edik szabásképben hányszor szerepel,

N_i az i -edik rendelés tételszáma,

Q_k az F_k -ből rendelkezésre álló darabszám,

L_h^j a j -edik szabásképhez tartozó alaptábla hossza,

$$\sigma_{kj} = \begin{cases} 1, & \text{ha } L_h^j = L_k \\ 0, & \text{egyébként.} \end{cases}$$

A célfüggvény a felhasználandó félkésztermékek összfelületére utal; a cél ennek a minimalizálása, mivel felületveszteségen az összes felszabandó alaptábla felületének és a leszabott rendelések felületének különbségét értjük. A rendelések összfelülete konstans, az alaptáblák szélessége is állandó, ezért vehettük fel ilyen alakban a célfüggvényt. Gyakorlati oldalról ez azt jelenti, hogy a többlettermelést hulladéknak vesszük,

azaz nem tároljuk a feleslegesen levágott darabokat.

5.3. A modell számítógépes megvalósítása

A matematikai alapok leírása után az alaptáblák, a rendelések legfontosabb jellemzőit és a leszabás konkrét üzemi feltételeit adjuk meg, majd a próbafeldolgozás tapasztalatait ismertetjük.

Az üvegyárban a két-asztalos szabásgéppel természetesen az alaptábláknak csak egy bizonyos osztályát szeretnék feldarabolni, amelyeknek fontos paraméterei összefoglalva az alábbiak:

- /i/ Az alaptáblák a minőségük szerint 4 osztályba sorolhatók:
A, B, C és D minőség /természetesen ez változhat/.
- /ii/ Az alaptáblák méretei jelenleg a következő értékek lehetnek:

vastagság	szélesség	hosszuság		
/mm/	/mm/	/mm/		
3	2000	2000,	2200,	2400
4	2000	2000,	2400,	2600
5-6	3000	2000,	2400,	2600,

A rendelések jellemzői közül a minőség és a vastagság nyilvánvalóan megegyezik az alaptáblákéval. A szélesség és a hosszúság 256-2000 mm között milliméterenként változhat és gyakor-

lati megfontolások miatt a tételszám legalább $50 / N_i \geq 50 /$.

A konkrét műszaki feltételek az alábbiak /a teljesség kedvéért a már említett feltételeket is megismételjük/:

- a/ A szabás két vágóasztalon történik, az elsőn az alaptáblát csikokra vágják fel, a másodikon a csikokat egyenként darabolják fel a kívánt méretre;
- b/ Az első asztalon a gép 9 vágófejjel rendelkezik, a második asztalon 12 vágófej van, de itt is csak legfeljebb 9 működhet egyszerre. Mindkét asztalon az aktív 9 vágófej közül az első rögzített, a többi állitható, ezért $S_1 = S_2 = 8$ /azaz legfeljebb 8 csik szabható egy alaptáblából és egy csikból legfeljebb 8 téglalap/;
- c/ A vágófejek egymástól legalább 256 mm-re vannak /ez a minimálisan levágható téglalap-méret/;
- d/ Egy alaptáblából legfeljebb csak 4 különböző méretű rendelést szabhatunk ki a rendelkezésre álló hely, a csomagolási és a raktározási gondok miatt, azaz $T = 4$;
- e/ A legyártott alaptáblák széleinek egyenetlensége miatt az a gyakorlat, hogy az alaptábláknak mind a négy oldalán az üveg vastagságától függően un. "selejtcsikot" vág-
nak:

3-4 mm vastagságra 35 mm,

5-6 mm vastagságra 50 mm széles a selejtcsik.

f/ Az első asztalra felhelyezhető alaptábla maximálisan 3000 mm széles és 3500 mm hosszú, a második asztalra felhelyezhető üveglap pedig maximálisan 2000 mm széles és 3000 mm hosszú lehet.

A következőkben ismertetjük az általunk alkalmazott módszer két lépését megvalósító számítógépes algoritmusok alap gondolatait. A két lépés a következő:

- /i/ az összes szabáskép elkészítése,
- /ii/ a nagyméretű ILP feladat megoldása.

5.3.1. Eljárás az összes szabáskép előállítására

Az összes szabáskép előállításának vázlata /a rendelésállomány egy csoportjára/ a 17. ábrán látható. Először a csikokat készítjük el. A z_i szélességű csik-vektorok képzéséhez a $z^i = z_i^1, z_i^2, \dots, z_i^M$ elemei közül kell az összes legalább elsőrendű és legfeljebb nyolcadrendű ismétléses kombinációt kiválasztani. Ezt a kombinatorikus problémát a következőképpen oldottuk meg, erősen kihasználva, hogy ez a részfeladat egy-dimenziós:

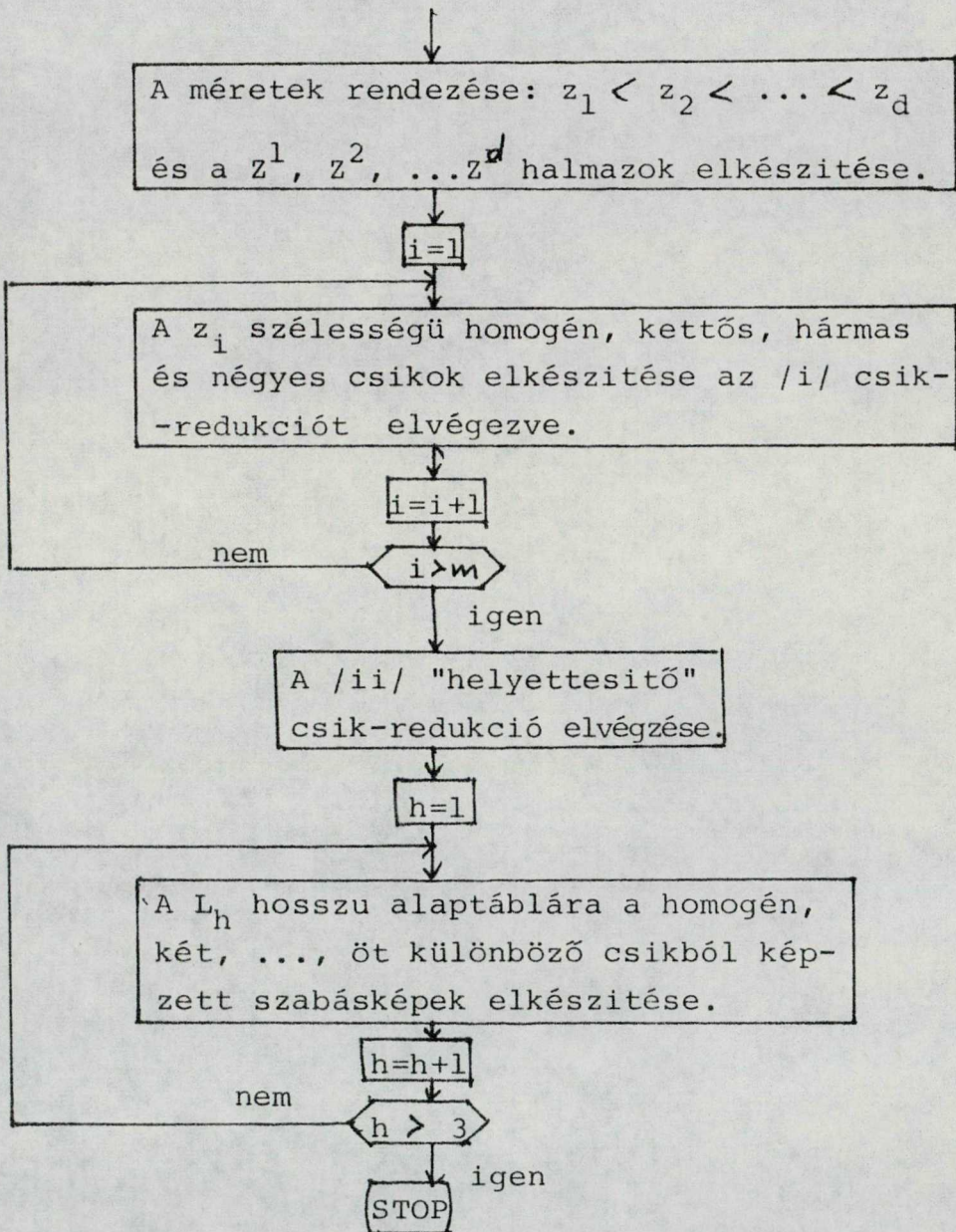
1./ Először a homogén csikokat készítjük el, ahol $c_{k_j^i}^i = \left[\begin{array}{c} W \\ z_j^i \end{array} \right]$

és az /i/ csik-redukciót is elvégezzük / $j = 1, 2, \dots, M$ /. /A csikok sorszámozásától eltekintünk, z_i^j a k_j -edik rendeléshez tartozik/.

Bemenet: F és R elemei;

Kimenet: szabás-vektorok, veszteségek és alaptábla-hosszak;

Megjegyzés: kettős csikon a $\sum_k \text{sign } c_{kj}^i = 2$ csikot értjük,
a hármas és négyes csik értelmezése hasonló.



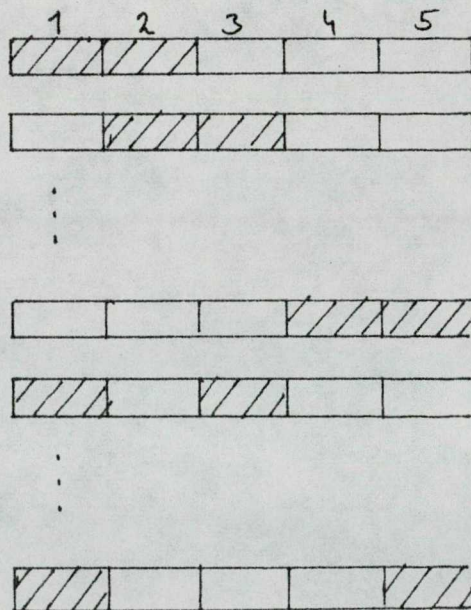
17. ábra. A szabásképek előállításának vázlata



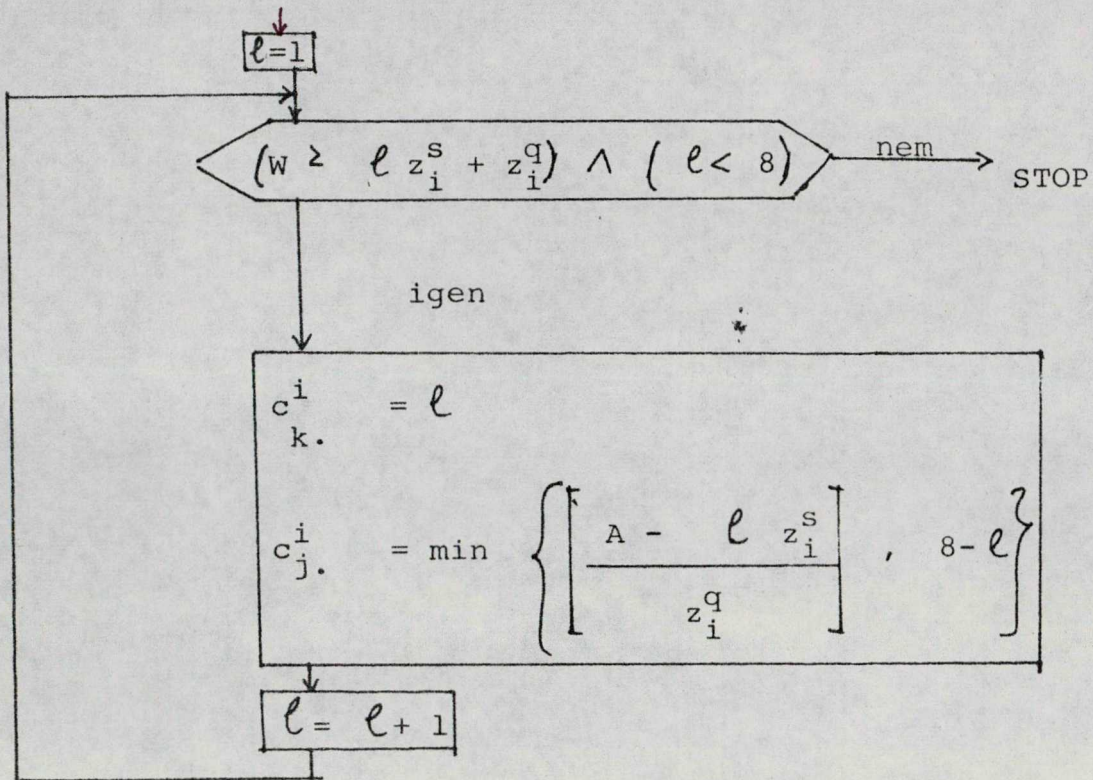
2./ A kettős csikok elkészítését két részletben oldottuk meg. Először az összes másodrendű kombinációt /az összes párt/ képezzük Z^i elemeiből, majd ezt felhasználva vizsgáljuk meg ebből a két elemből képzett legfeljebb nyolcadrendű ismétléses kombinációkat /mindig szerepel mindkét elem a kombinációban/.

Pl. legyen $M=5$, ekkor a párok kiválasztását a 18. ábra szerint végezzük el, ahol a vonalkázott rész a pár tagjait jelöli. Legyen egy ilyen pár z_i^q és z_i^s / $z_i^q < z_i^s$ / és tegyük fel, hogy z_i^q a j -edik z_i^s a k -adik rendeléshez tartozik, a belőlük előállítható csikvektorok képzését a 19. ábra mutatja.

3./ A hármas és négyes csikok elkészítése hasonló módon történik, mint a kettős csikoké.



18. ábra. A másodrendű kombinációk kiválasztása



19. ábra. Kettős csikok képzésének blokkdiagramja.

Az algoritmus alapján világos, hogy az összes lehetséges csik-vektort előállítjuk az előbb vázolt módon. Ez az előállítás túl hosszúnak tűnik, mivel négy blokból álló programot kellett írni. Miért választottuk mégis ezt az utat? Legfontosabb szempont az volt, hogy a rendelések megközelítőleg egyforma gyakorisággal szerepeljenek a csikokban, bármikor is szakítódik meg a csik-vektorok képzése amiatt, hogy már tulságosan sok vektor készült el, azaz ha a számítógépben lefoglalt terület megtelt.

4./ Az összes csik-vektor elkészítése után a /ii/ "helyettesítő" csik-redukciót vizsgáljuk meg. A végrehajtása időigényes, ezért a képletet egyszerűsített formában vesszük figyelembe.

A kettős csikoknál /23/ helyett csak a

$$v_1^i \geq \frac{c_{11}^i}{c_{12}^i} \cdot v_2^i + \frac{c_{21}^i}{c_{23}^i} \cdot v_3^i$$

képletet nézzük, a hármas és a négyes csikoknál is hasonlóan járunk el. Mivel a csikok a szélességük szerint monoton nem-csökkenő sorrendben vannak, ezért hátulról visszafelé haladunk. A helyettesítést kereső eljárás az egy fa "endorder" bejárásához hasonló, ahol a fa gyökere a helyettesítendő csik. Egy út az egy lehetséges helyettesítés. A balrészfán lemegyünk a levélig és onnan megyünk balról-jobbra végig a leveleken /itt csak a leveleket keressük meg/.

5./ A szabásképeket a csikok elkészítéséhez hasonlóan állítjuk elő. Először az egyféle, majd a két-, három-, négy- és ötféle csikokat tartalmazó szabásrajzokat keressük meg a L_1 hosszú alaptáblára, azután a L_2 hosszú alaptáblára, stb. A legalább hatféle csikot tartalmazó szabásképeket nem készítjük el, mivel kis valószínűséggel fordulnak elő. A szabásképek előállításánál a /25-28/ feltételeket kellett figyelni, és azt, hogy kisebb alaptáblából ne legyen leszabható, azaz kétszer ne állítsuk elő.

5.3.2. Az ILP feladat megoldása

Az ILP feladat megoldására a VOPP programkönyvtárban levő VDG3 nevű vegyes-egészértékű programcsomagot választottuk ki, amely DOS-ban futtatható. Az együtthatómátrix elemeinek a száma legfeljebb 8192 lehet /ez a FORTRAN tömb-méret miatt ennyi/ és legfeljebb 400 lehet a sorok ill. az oszlopok száma. A VDG3 részletes leírása, a szükséges környezet ismertetése [78] -ban megtalálható, ezért itt nem térünk ki erre.

5.4. Futtatási eredmények

A következőkben a gyártól kapott rendelésállomány két csoportjának próbafeldolgozását ismertetjük, összevetve a számítógépes eljárás optimális szabástervét a hagyományos "sorozatvágás" és egy heurisztikus "mohó" algoritmus eredményeivel. A futtatásokat a József Attila Tudományegyetem Kalmár László Kibernetikai Laboratóriumának R-40 számítógépén végeztük el.

A csoportokat alkotó rendelések adatai a 20. ábrán láthatók. A méreteket centiméterben adjuk meg. Mindkét csoport 14 rendelésből áll. Erre a két csoportra a hagyományos szabási módszer felületvesztesége:

- az 1. csoportnál: 45,6 %,
- 2 2. csoportnál: 40,1 %.

Természetesen az alaptábla szélén levágandó "technológiai" selejtcsikot is mindig beleszámítjuk a felületveszteségbe.

Vastagság: 3 mm,

minőség: C

Vastagság: 3 mm,

minőség: B

Azonosító	Méretek	Tételszám	Azonosító	Méretek	Tételszám
A1	134 x 74	420	B1	152 x 80	220
A2	50 x 74	207	B2	184 x 84	700
A3	134 x132	300	B3	100 x 100	200
A4	50 x132	150	B4	162 x 78	300
A5	132 x102	204	B5	130 x 82	150
A6	50 x102	170	B6	134 x 73	280
A7	132 x132	204	B7	50 x 74	138
A8	86 x132	204	B8	134 x 132	200
A9	84 x 84	204	B9	50 x 132	100
A10	134 x 44	204	B10	132 x 102	136
A11	150 x134	80	B11	132 x 132	136
A12	96 x 85	80	B12	86 x 132	136
A13	82 x 82	80	B13	84 x 84	136
A14	150 x 44	80	B14	134 x 44	136

1. csoport

2. csoport

20. ábra. A rendelésállomány két csoportja

Amikor a szabásképek számát felső korláttal becsültük, ez a szám olyan nagy lett, hogy emiatt először heurisztikus megoldással próbálkoztunk. Az ún. "mohó" algoritmust alkalmaztuk:

a rendeléseket a nagyobb méretük szerint nem-növekvő sorrendbe rendeztük és előlről elindulva próbáltuk elhelyezni őket az alaptáblákon. Csak a homogén csikokat engedték meg, azaz egy-dimenziós volt a feladat. Az algoritmus gyors, a gépidő és a tárigény elenyésző, de a gyártól kapott próbaadatokkal futtatva a hagyományos szabáshoz képest átlagosan 6-9 %-kal csökkent csak a felületveszteség. Pl. az 1. csoportnál 37 % lett, azaz a sorozatvágáshoz képest 8,6 %-kal csökkent, míg a 2. csoportnál 6,9 % a megtakarítás.

A VDG3 program-adatok méretei miatt egy csoporton belül legfeljebb 20 rendelés optimalizálását láttuk célszerűnek amit a gyártól kapott adatok is igazoltak, mivel egy csoportban húsznál több rendelés elvétele fordult csak elő - így 400 szabásképet tudunk figyelembe venni. Az 1. csoportnál a csikok száma 26 lett, ezekből a csikokból 862 szabásképet kaptunk /az /i/ és /ii/ szabáskép-redukciók végrehajtásával ez 650 alá csökkenthető/. Felmerült a probléma, hogyan válasszuk ki a legfeljebb 400 szabásképet? A próbafeldolgozásnál ezt heurisztikusan végeztük el azon elv alapján, hogy a kiválasztott szabásképek a legkisebb veszteségűek legyenek és a közel 400 szabásképben a rendelések gyakorisága közel egyenlő legyen. Így 395 szabásképet kaptunk /a raktárt figyelmen kívül hagytuk/, a VDG3 ezekből állította össze az optimális szabástervet, amely a 21. ábrán látható a 2. csoport optimális szabástervével

együtt. A szabás-vektor i . komponense a beolvasás sorrendjében i . rendeléshez tartozik, a nullákat nem tüntettük fel /a szabás-vektor itt sorvektor/. Pl. az S1 szabás-vektort 200 x 200 cm-es alaptáblára kell 204-szer alkalmazni és a szabásmintában az A1, A2, A3, A10 jelű rendelések mindegyike egyszer szerepel. Másrészt az A1 rendelés az S1, S2 és S3 szabásképekben kerül leszabásra.

A szabásképek előállításának gépideje 2-3 perc csoportonként az R-40 számítógépen, a VDG3 futási idejére nehéz becslést adni, mivel a folytonos optimum megtalálása után változó számú transzformációt hajt végre, amíg a megadott határon belül levő egészértékű megoldást eléri. Pl. az 1. csoportnál 21 transzformációt végzett el. Az egészértékű megoldás a folytonos optimumtól 0,7 %-kal tért el, ez az eltérés az alkalmazás szempontjából lényegtelen. A szabás-vektorokból a szabásképek elkészítése egyszerűen elvégezhető.

	Sorozatvágás	"Mohó" alg.	ILP
1. csoport	45,6 %	37,0 %	21,7 %
2. csoport	40,1 %	33,2 %	23,4 %

22. ábra. A felületveszteségek összevetése

A rendelések "folyamatos" kielégítése felmerült Orosházán is, azaz, ha egy rendelés szerepel egy szabásképben, akkor addig szerepeljen a soron következőkben, amíg ki nincs elégítve.

Azonosító	Hány db	A szabás-vektor														Alaptábla
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	
S1	204	1	1	1						1						220
S2	96	1	1	1	1											220
S3	122	1	1		1			1								220
S4	32				1			1	1							240
S5	50				1			1		2						240
S6	52									2	2					240
S7	34					1				2						240
S8	85					2	2									220
S9	40											1	2		1	240
S10	40											1		2	1	240

1. csoport. Az optimális szabásterv felületvesztesége:
21,7 %.

Azonosító	Hány db	A szabás-vektor														Alaptábla
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	
T1	1													4		200
T2	75				1	2										220
T3	51				1					1		1				220
T4	85	1								1		1				220
T5	135	1							1	1						220
T6	66								1					2	1	240
T7	71				1		2								1	220
T8	69		1					2	2							240
T9	232		2													200
T10	167		1	1												200
T11	33			1						1		1				240
T12	104				1					1		1				220

2. csoport. Az optimális szabásterv felületvesztesége:
23,4 %.

A lineáris programozással kapott optimális szabásminták nem tesznek eleget a folyamatosságnak, a szabásképek csak "majdnem folyamatos" sorrendbe állíthatók. A 21. ábrán már ilyen sorrendben vannak, az 1. csoportnál csak egy rendelés leszabása szakítódik meg, és az is csak egyszer. A 2. csoportnál, ha a T1 szabásképp helyett a T6-t nem 66-szor, hanem 68-szor alkalmazzuk, akkor csak egy rendelés leszabása szakítódik meg kétszer. Mindkét sorrend a gyakorlatban megfelelő, mivel egy rendelés tárolására van hely. Ezt a két sorrendet próbálgatással kaptuk.

Az előbbieken vázolt gondolatok alapján a JATE Kalmár László Kibernetikai Laboratórium egy munkacsoportja 1979-ben - dr. Csirik János vezetésével - elkészített egy programrendszert az orosházi üvegyár részére.

6. A GGM gyakorlati megvalósítása

Az alábbi fejezetben ismertetjük a GGM egy lehetséges realizálását, az elkészült programrendszer elvi alapjait, amelyet KALMÁR JÁNOS-sal közösen készítettünk el. Kalmár János önálló munkája a készletfeltételt figyelembe vevő heurisztikus eljárás megoldása, a számítógépes rutinok megírása és letesztelése, valamint a 6.1. alpont végén megadott /31/ képlet beépítése a programrendszerbe /amelyre ebben a dolgozatban nem térünk ki/. Egy-dimenziós és két-ütemű két-dimenziós quillotine-vágásokra külön készült egy-egy /elveiben, felépítésében hasonló/ rendszer az adatbeolvasás és az adatkezelés eltérő jellege miatt. A szimplex-táblázat inicializálásánál a raktárkészlet figyelembe vételére készítettünk egy heurisztikus eljárást, amely a kétfázisú módosított szimplexmódszer első lépését végzi el. Ehhez hasonlóval a szakirodalomban nem talákoztunk, habár egyszerű ötleten alapszik. A fejezet többi részében a programrendszer jellemzőit mutatjuk be, majd futtatási eredményeket ismertetünk.

6.1. A készletfeltétel kezelése

A 2.1.1. alpontban bemutattuk, hogy a darabolás lineáris programozási feladathoz hogyan csatolható a raktárkészleten levő alapanyagok feltétele



$$/7/: \quad x_j \geq 0 \quad /j = 1, 2, \dots, n/$$

$$\sum_{j=1}^m a_{ij} x_j \geq N_i \quad /i = 1, 2, \dots, m/$$

$$\sum_{j=1}^m \sigma_{kj} x_j \leq Q_k \quad /k = 1, 2, \dots, p/$$

$$\sum_{j=1}^m c_j x_j \rightarrow \min !$$

ahol $\sigma_{kj} = \begin{cases} 1, & \text{ha a } j\text{-edik szabáskép a } k\text{-adik alapanyagból} \\ & \text{készült,} \\ 0, & \text{egyébként.} \end{cases}$

A kibővített feltételrendszer kezelése látszólag nem ütközik akadályba, a GGM a 2. fejezetben leírtak szerint alkalmazható /2.1.1. alpont/. Akkor van csak probléma, ha nem tudunk egyszerűen megadni homogén szabásképekből egy induló bázismegoldást, azaz ha a kétfázisú módosított szimplexmódszer első lépése nem triviális. Mikor fordulhat ez elő? Akkor, ha sokféle alapanyag van a raktáron, de viszont mindegyikből kevés darab és ekkor egy rendelés egyféle alapanyagból nem elégíthető ki. Ilyen esetben már az induló bázismegoldás is közel van az optimálishoz, ezért nehéz megadni. Másrészt olyan bázismegoldásra van szükségünk, amelynél a bázisvektorokból álló mátrix inverze könnyen és numerikusan pontosan kiszámolható. Egy nagyméretű mátrix invertálásakor a kerekítési hibák miatt lehet, hogy a tábla a további számolásokra alkalmatlanná válik. Így inicializáláshoz

a legcélszerűbbnek továbbra is a homogén szabásminták készítése és a belőlük előálló diagonális mátrix tűnik.

Az általunk elkészített heurisztikus eljárás homogén szabásminták létrehozásán alapszik. Alapgondolata az, hogy a túl nagy tételszámu rendeléseket részrendelésekre /fiktív rendelésekre/ bontja, amelyek külön-külön egy-típusu alapanyagból már kielégíthetőek. Így /a szimplex táblázat méreteinek növelése árán/ az induló bázismegoldás mátrixa diagonális lesz és a szimplex tábla a szokott módon inicializálható. Megjegyezzük, hogy a tábla méretének növelése a hátizsákfeladat megoldásának időszükségletét nem növeli /ami becslések szerint a teljes számítási idő 80 %-a [76] /, mert a feladat hossza tulajdonképpen nem változik. Ugyanis a 2.3. lemma szerint /2.2. alpont/, ha a részrendelések hossza egyenlő, akkor közülük elegendő azt megtartani, amelynek árnyékköltsége nagyobb /azaz, ha $e_i \geq e_j$ és $\pi_i \leq \pi_j$, akkor $a_i = 0$ /.

Az eljárás leírásának segédeszközei: rendezzük át az alapanyagokat az $L_1 > L_2 > \dots > L_p$ és a rendeléseket a $e_1 \geq e_2 \geq \dots \geq e_m$ sorrendbe. Definiáljuk a D szabásmátrixot ill. az E költségmátrixot:

$$D = \begin{bmatrix} d_{ij} \end{bmatrix} \quad \begin{matrix} i = 1, \dots, m \\ j = 1, \dots, p \end{matrix}$$

ahol d_{ij} = aj-edik alapanyagból leszátható i-edik rendelések száma /pl. egy-dimenziós esetben $d_{ij} = \begin{bmatrix} L_j \\ e_i \end{bmatrix}$ írja le a megfelelő homogén szabásképet/;

$$E = \begin{bmatrix} e_{ij} \end{bmatrix} \quad \begin{matrix} i = 1, \dots, m \\ j = 1, \dots, p \end{matrix}$$

$$\text{ahol } e_{ij} = \begin{cases} (L_j - \ell_i d_{ij}) / d_{ij}, & \text{ha } d_{ij} > 0 \\ \infty, & \text{egyébként} \end{cases}$$

Ekkor egy homogén szabásmintákból álló induló bázismegoldást a következő ILP szolgáltatja:

$$x_{ij} \geq 0, x_{ij} = \text{egész} \quad /i = 1, 2, \dots, m; j = 1, 2, \dots, p/$$

$$/30/ \quad \sum_{j=1}^p d_{ij} x_{ij} \geq N_i \quad /i = 1, 2, \dots, m/$$

$$\sum_{i=1}^m x_{ij} \leq Q_j \quad /j = 1, 2, \dots, p/$$

$$\sum_{i=1}^m \sum_{j=1}^p e_{ij} d_{ij} x_{ij} \longrightarrow \min$$

Tehát a célfüggvény most közvetlenül a fajlagos felületveszteséget minimalizálja és nem a költséget.

Vegyük észre, ha a /30/ képletekből a d_{ij} együtthatókat elhagyjuk és az egyenlőtlenségek helyett egyenlőséget írunk, akkor a közismert szállítási feladat áll előttünk. Ezért annak a megoldására alkalmazott disztribúciós módszer kezdő megoldását szolgáltató heurisztikus eljárás egy módosításával határozzuk meg az x_{ij} értékeket a következőképpen:

$i = 0$

1. lépés $i = i+1$
 $x_{ij} = 0 \ /j=1,2,\dots,p/$

2. lépés $j = \text{az az index, amely } e_{ij} \text{ minimális;}$
~~2. lépés~~ $\text{if } e_{ij} = \infty \text{ then goto 3. lépés;}$
 $e_{ij} = \infty$

$x_{ij} = \min \left\{ Q_j, \left[(N_i + d_{ij} - 1) / d_{ij} \right] \right\}$
 $Q_j = Q_j - x_{ij}$
~~3. lépés~~ $N_i = N_i - x_{ij} \cdot d_{ij}$
 $\text{if } N_i > 0 \text{ then goto 2. lépés;}$

3. lépés $\text{if } i < m \text{ then goto 1. lépés;}$
 end

Látható, hogy ez az eljárás a legkisebb selejtű homogén szabásképeket használja fel minél többször, az aktuális igény és a készlet figyelembe vételével. A szimplex-táblázat ezután a D és az $X = \left[x_{ij} \right]_{\substack{i=1,\dots,m \\ j=1,\dots,p}}$ mátrixok ismeretében feltölthető, a részrendelésekre bontás elvégezhető. Ez a heurisztikus eljárás már ezeket a fiktív rendeléseket létre is hozta, mivel egy rendelést annyi fiktív rendelésre bontunk, ahány alaptábla-féleség kellett a homogén szabásképekhez, azaz $\sum_j \text{sign } x_{ij}$ -féle részrendelésre.

A készlet-feltétel fenti kezelése csaknem dimenzió-független, tehát két-dimenziós esetben is a fenti eljárás alkalmazható kis módosítással. Ugyanis csak a d_{ij} és az e_{ij} kiszámítása változik.

Ha az alapanyagon a rendelés nem forgatható el, akkor

$$d_{ij} = \begin{bmatrix} L_j \\ e_i \end{bmatrix} \cdot \begin{bmatrix} W_j \\ w_i \end{bmatrix},$$

egyébként

$$d_{ij} = \max \left\{ f_1 \cdot \begin{bmatrix} L_j \\ e_i \end{bmatrix} + f_2 \cdot \begin{bmatrix} L \\ w_i \end{bmatrix} \right\}$$

ahol $w_i \cdot f_1 + e_i \cdot f_2 \leq W_j$; $f_1, f_2 \geq 0$, egész,

tehát itt egy hátizsákfeladat megoldásaként adódik a homogén szabáskép /ahol kétféle homogén csikot is figyelembe veszünk egy szabásképben/.

Az e_{ij} pedig a következő:

$$e_{ij} = \frac{(L_j w_j - e_i w_i d_{ij})}{d_{ij}}, \text{ ha } d_{ij} > 0$$

∞ egyébként

Ha az eljárás által kapott homogén szabásképek nem alkotnak bázismegoldást, akkor a raktár-feltételek lazítása a célszerűbb /elég egyféle alapanyag mennyiségét lényegesen megnövelni/.

Megjegyezzük, hogy a darabolási feladat folytonos megoldásának egészértékűvé tétele, felfelé kerekítése "megsértheti" a raktárkészlet-feltételt. A gyakorlatban több megoldás lehet. Ha "puha" a korlát /azaz kis büntetéssel átléphető/, akkor utólag még rendelünk a megfelelő alapanyagból. Ha "kemény" a feltétel, akkor a hiányzó alapanyagokra készült szabásminták rendeléseit "kézi összeillesztéssel" is le lehet szabni valamelyik raktáron levő félkésztermékből. Javasolható még az alapanyagok tételszámának csökkentése is, azaz

$$/31/ \quad Q_k = P_k - \min \left(m + p, \sum_{j=1}^m \delta_{kj} \right)$$

Itt komoly problémát jelent, hogy a GGM alkalmazásakor egyidejűleg csak a bázisban szereplő szabás-vektorok ismertek, nem a teljes feltételrendszer. A /32/ képlet következetes alkalmazásából viszont az következik, hogy minden bázisvektor-csere a /2/ feladat jobb oldalának módosulását is jelenti /a bázisból kikerülő szabáskép alapanyagának virtuális készlete 1-el nő, míg a bekerülő szabáskép készlete 1-el csökken/. Ennek figyelembevétele a generáló-elem kiválasztásánál létfontosságú, amely gondos programozással megoldható.

6.2. A GGM-t realizáló programcsomag programterve

A 2. fejezetben leírtak alapján készítettük el a GGM egy lehetséges programrendszerét. A szimplex-táblázat inicializálásánál a 6.1. alpontban leírt eljárást alkalmazva a raktárkészlet-feltételeket is figyelembe vettük.

A hátizsákfeladatok megoldására a 2.2. pontban ismertetett lexikografikus rendezésen alapuló leszámláló algoritmust használtuk, amelybe a vágófejek számát és a gyorsító feltételeket is beépítettük. A gépidő lényeges csökkentését értük el azzal, hogy a lineáris programozási feladat célfüggvényének változását minden báziscsere-transzformációnál figyeltük és ha a célfüggvény egy előre megadott korlátnál kevesebbet csökkent, akkor "elfogadtuk" a megoldást.

A számítógépes eljárás mindegyik dimenzióban 5 jól elkülöníthető részre bontható, melyeket célszerű külön megírni és tesztelni. A programok COMMON mezővel és disc-file segítségével kommunikálnak egymással /utóbbi tartalmazza a mindenkori bázis szabás-vektorait, melyeket egyébként csak a szimplex-táblázat megfelelő részletének invertálása útján kaphatnánk vissza; két-dimenziós esetben az egyes szabásképek csik-összeállítását is itt tároljuk/.

Az input szerkezete: egy-dimenziós eset,

a. Vezérlő-kártya:

EPS	- a konvergencia-ellenőrzést biztosító paraméter,
KES	- a használható vágófejek maximális száma.

b. Alapanyag adat-kártya /utolsó blank/:

AR	- az alapanyag egységára,
KESZL	- a készlet darabszáma,
HOSSZ	- az alapanyag hossza.

c. Megrendelés adat-kártya /az utolsó blank/:

TSZAM - az igényelt mennyiség,
THOSSZ - a rendelés hossza.

két-dimenziós eset

a. mint fent.

b. kiegészül az alábbi paraméterrel:

SZEL - az alapanyag szélessége.

c. kiegészül az alábbi paraméterekkel:

TSZEL - a tétel szélessége,

TFORG - a tétel forgathatóságát jellemzi.

Az eljárás fázisai:

1. fázis: az előbb leirt szerkezetű input-adatok beolvasása, durva adatellenőrzés, pl. nem nagyobb-e a rendelés-állomány az alapanyag-készletnél?

2. fázis: homogén szabásképeket tartalmazó szuboptimális megoldás keresése a /30/ feladat heurisztikus megoldásával.

3. fázis: a bázis- /szabás-/ vektorok disc-file-ra írása, a szimplextábla inicializálása az előző fázis bázismegoldása alapján, esetleges részrendelésekre bontás.

4. fázis: iteráció. A "legjobb" szabás-vektor meghatározása a hátizsákfeladatok megoldása útján. Ha a segédfeladat célfüggvénye negatív, g o t o 5. fázis! Egyébként szimplextransz-

formáció, bázisvektor-csere, a disc-file karbantartása. Ha a célfüggvény EPS százalékkal kisebb mértékben javul, g o t o 5. fázis! Egyébként g o t o 4. fázis!

5. fázis: a disc-file-on található megoldás listázása, ill. két-dimenziós esetben a szabásképek kirajzolása.

Interaktív hozzáférés esetében a 4. fázisba be lehet építeni EPS értékének újra-definiálását. Ennek korlátozott futási idő esetén van jelentősége, ha nem tudjuk előre becsülni a számítási időt. Ekkor nagy EPS-el, pl. 10 %-kal kezdünk, s amint elakad az algoritmus, egyre kisebb EPS-el indítjuk újra, ha van még rá idő, így a számolt részeredmény sem veszik el.

A megoldandó feladat méretétől /rendelések és alapanyagok száma/ függ a tárigény. Az általunk megvalósított szabásoptimalizáló programrendszer helyfoglalása a bináris kódon túl még

$$2 / m + p / 2 + mp + 12 m$$

szó /1 szó = 1 egész szám/, ahol m a rendelések száma, p pedig az alapanyagé. A két-dimenziós programrendszer helyszükséglete ugyanennyi, mivel disc-en tároljuk a csikok és szabásképek jellemzőit.

A számítási idő - az ismerttetett gyorsító módszerek együttes alkalmazásával - nagyobb méretek mellett is csak néhány perc, amit felerészben a gyakori disc-hez fordulások okoznak/ a szabás-vektorokat ott tároljuk, hogy a CPU-ban csak a szimplextábla legyen/.

6.3. Futtatási eredmények

Az egy- és a két-dimenziós programrendszerre egy ill. két jellemző futtatást ismertetünk. Az egy-dimenziós input adatokat a 23. ábra tartalmazza /a rendelések maximális száma: 50, az alapanyag-féleségeké: 10 lehet/. A konvergencia-faktort $EPS=1,0$ -nak vettük és nem kellett módosítani az optimális megoldás eléréséhez. A heurisztikus induló bázismegoldás /amely homogén szabásképekből áll/ elkészítése során egyuttal a lehetséges megoldások halmazát is megvizsgáljuk, nem üres-e? A báziscsere-transzformációk száma az eddigi futtatások alapján a rendelések számától függ elsősorban, kb. $1,5xm$.

A 24. ábrán láthatók az optimális szabás-vektorok /a számuk: 16/. Ha a hulladékot viszonyítjuk a felhasznált össz-alapanyaghoz, akkor a felületvesztesség:

a heurisztikus bázismegoldásnál:	6,3 %
az optimális megoldásnál	: 3,3 %

Mivel az alapanyagok hosszával nem arányos a költségük, ezért a legelőnyösebb 260 cm-es alapanyagot kell feldarabolni /amelyből volt bőven a raktáron/. A gépidő a Hewlett-Packard számítógépen 1-1,5 perc a hasonló felépítésű rendelésállományokra.

A két-dimenziós próbafuttatást az orosházi üveggyár 20. ábrán közölt rendelésállományára is elvégeztük. Az optimális

Egy-dimenziós darabolási feladat

Az alapanyagok paramétereit:

költség Ft	tételszám db	hossz cm
15	300	150
16	400	180
18	400	210
20	300	230
22	10000	260

A rendelések paramétereit:

sorszám	tételszám db	hossz cm
1.	24	80
2.	48	72
3	60	124
4.	20	38
5.	49	46
6.	54	54
7.	62	45
8.	70	77
9.	70	112
10.	35	120
11.	44	102
12.	50	94
13.	45	78
14.	45	72
15.	75.	64
16.	80	68

A vágófejek száma: 5.

A konvergencia-faktor = 1,0.

A heurisztikus bázismegoldás:

alapanyag- hossz	db	rendelés sorszáma
150	24	2.
150	23	14.
180	16	7.
210	4	4.
210	22	11.
210	25	12.
210	27	16.
230	10	5.
230	14	6.
230	35	9.

alapanyag hossz	db	rendelés sorszáma
260	8	1.
260	30	3.
260	24	8.
260	18	10.
260	15	13.
260	19	15.

23. ábra. Az input adatok és a heurisztikus bázismegoldás

Alapanyag- hossz	db	A szabás-vektor komponensei															
		1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	13.	14.	15.	16.
260	1		1					1	1								1
260	20				1				2								1
260	6											2			1		
260	35									1					1		1
260	17					3		1	1								
260	10	1							1			1					
260	8			1				3									
260	5							1	1							1	1
260	2	1	1				2										
260	14	1								1							1
260	19							1	1		1						
260	39						1		1			1					
260	17										1		1				
260	47		1	1													1
260	13						1							1		2	
260	6			1													2

A célfüggvény értéke: 5.698.

A felületvesztés : 3,3 %.

24. ábra. Az optimális szabás-vektorok

20. ábra.

Az orosházi rendelésállomány első csoportja

Azono- sitő	hány db	szabás-vektorok														alaptábla hossza	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14		
Z1	34	1				1			1								220
Z2	3	1	1		1			1									220
Z3	96	1	1		1												220
Z4	85					2	2										220
Z5	16							1	1								240
Z6	204	1	1	1							1						220
Z7	38	1	1					1									220
Z8	52								3								240
Z9	102				1			1		2							240
Z10	5	1	1		1			1									220
Z11	40	1	1									1			1		240
Z12	40												2				240
Z13	40											1	2	1			240

A csoport felületvesztesége: 22,4 %.

Az orosházi rendelésállomány második csoportja.

Azono- sitő	hány db	szabás-vektorok														alaptábla hossz	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14		
V1	74	3															240
V2	297		2														200
V3	50		2						1							1	240
V4	12			2	1												220
V5	36				1							1					220
V6	50					3											240
V7	20		1	2													220
V8	63							1	1	1						1	220
V9	88							1		1							240
V10	100				1						1		1				220
V11	16				1			2								1	240
V12	136				1							1		1			220
V13	8							3								1	220
V14	68			2												2	220
V15	38		1					2	2								240

A csoport felületvesztesége: 22,9 %

25. ábra. Az orosházi rendelésállomány próbafuttatásának optimális szabás-vektorai.



szabás-vektorokat a 25. ábra tartalmazza, így a 21. ábra optimális szabásmintáival összevethetők. A két megoldási módszerből adódik az eltérés, mivel a GGM az a folytonos megoldást kerekíti fel egészértékűre, míg a másik az eleve ILP-t old meg. Másik oka lehet az eltérésnek az, hogy a GGM-nél a célfüggvény változását figyeljük /előre megadott küszöbszámmal/, így még az optimális megoldás előtt megállhat a program.

A gépidő két-dimenziós esetben jelentősen megnő, mivel a viszonylag kis központi memória miatt mágneslemezen tároljuk a szabás-vektorokat és az őket alkotó csik-vektorokat. Az adatok mozgatása pedig sok időt igényel. A programrendszer felépítése egyébként ugyanaz, mint az egy-dimenziós esetben.

7. Egyéb megközelítések

A következőképpen röviden vázoljuk az előbbieken ismertett három megközelítéstől eltérő, általában speciális feladatok megoldására készített eljárásokat.

Az egy-dimenziós feladatok megoldása iránt az igény már az ötvenes években jelentkezett. EISEMANN a szabásképeket két szabásgép egyenletes leterhelésére lineáris programozási feladat optimális megoldásaként adta meg [25]. Miután a GGM ismertté vált, általában csak olyan speciális üzemi, műszaki feltételekre készítettek algoritmusokat, amelyek lineáris programozási feladatként nem írhatók fel, ill. nehezen oldhatók meg. STANTON acélrudak feldarabolását mutatja be [102], ahol a keletkező hulladék újrahasznosítását, valamint a rendelés leszabásakor a szükséges tárolás, kezelés költségeit, a darabológépet, a készletanyagokat is figyelembe veszi. A feladat megoldására heurisztikus eljárást adott.

TILANUS és GERHARDT acéltekercsek nyirását ismerteti [104]. A rendelésállományból az együtt levágható rendeléseket kiválasztották, amelyekből az egy szabásképeben leszabható hat rendelést kiemelték. Az eljárásuk két lépésből áll és dinamikus programozáson alapszik. A szabásminták elkészítésénél a rendelések határidejét, a felvágandó tekercs minőségét és a rendelések kielégítésének folyamatosságát is figyelembe vették.

COFFIELD és CRISP szintén tekercek nyírását írta le [15], amikor a határidők szerint a rendelkezéseket súlyokkal látták el /periódusokba osztották/. EISEMANN alapgondolatát használták fel, nagyméretű lineáris programozási feladatot oldottak meg. LITTON szőnyegek feldarabolásáról számol be [81], ahol a kielégítendő rendelkezések száma kevés. Megoldásként statisztikai /Monte-Carlo/ módszereket alkalmazott.

Az un. láda pakolási /bin packing/ feladatok megoldásával többen foglalkoztak. Legalapvetőbb a JOHNSON-DEMERS-GAREY-GRAHAM szerzők eredménye egy-dimenziós esetben, akik több /pontosan 4/ heurisztikus szekvenciális algoritmust ismertettek és a "legrosszabb" esetekben megadták az optimális megoldástól való eltérést is [60], [82]. Azóta több újabb, elsősorban elméleti eredmény született ebben a témakörben [8], [73], [10].

EILON és CHRISTOFIDES a csomagolási, pakolási problémára két eljárást adott. Az egyik 0-1 programozáson alapszik, a másik pedig egy heurisztikus algoritmus. Az ismert paraméterű ládák /dobozok/ felhasználását minimalizálták [24]. Eredményeik nagymértékű továbbfejlesztése található [13]-ban.

Elsősorban elméleti megközelítésnek számít az un. szubgradiens optimalizáció /pl. HELD-KARP-CROWDER ismerteti [52] /, amely a lineáris programozási feladat DANTZIG-WOLFE-féle dekompozíciós felbontásán alapul.

Gyakori két-dimenziós esetet tárgyal HAHN [50] , amikor olyan alapanyagok feldarabolására ismerttet egy dinamikus programozást felhasználó eljárást, amelyek hibát, hasznosíthatatlan részt tartalmaznak. Erre a feladatra utalás történt már [35] -ben is. Érdekes alkalmazás található [16] -ban, ahol keményfa lapokból /amelyek szabálytalan alakúak/ puskatusokat vágnak ki. Az alapanyagot téglalappá egészítik ki, ahol a hasznosíthatatlan részeket is téglalapokkal fedik le, míg a puskatusok trapéz alakúak.

PEGELS két heurisztikus algoritmust mutat be [94] , amelyek a veszteséget, a rendelések elrendezését és az alkalmas szabásgépet veszik figyelembe. Az első eljárás a kézi szabást próbálja heurisztikusan kiterjeszteni, míg a második szisztematikusan keresi az előnyös veszteségnek eleget tevő szabásképet. FILMER ezt a két-dimenziós feladatot heurisztikusan oldotta meg, bizonyos kulcs-rendelésekhez kereste meg a legjobb összeállítású szabásképet [28] .

A szabásképek már említett folyamatosságára, rendezésére próbált HANXMAN megoldást adni [54] , [55] üvegtáblák két-ütemű feldarabolásakor. NILSSON eredményei alapján [92] szekvenciálisan készítette el a szabásképeket, ahol a rendezőelv a sürgősség, az alaptáblák vagy rendelések folyamatossága, stb.

A darabolás határterületei érinti PFEFFERKORN két-dimenziós heurisztikus elhelyező algoritmusá, amellyel szobák bu-
torozását oldotta meg [95] . Mig GARFINKEL egy helyiség ta-
pétázása esetén a hulladék, a minta és a kiszabás viszonyát
elemezte [31] .

A heurisztikus módszerek nagy előnye éppen abban rejlik,
hogy a környezet nem-formalizálható jellemzőit figyelembe vé-
ve adnak elfogadható szabásmintákat /pl. IRONMMAN a humán
tényezők kiemelkedő fontosságát hangsúlyozza [59] /. Sőt CLOUD
szerint - aki a hullámlemezgépek néhány méret és vágási jelleg-
zetességét írta le - a tisztán elméleti programok a manuálisak-
nál nem jobbak [14] , hasonlót jelentett ki SCHAFFER kis meny-
nyiségű tekercsek feldarabolása kapcsán [100] . A darabolási
eljárásokat kidolgozók közötti nézeteltérésekre jellemző
COVERDALE és WHARTON [18] , valamint STANTON vitája [103] .

A témakör áttekintésére ketten vállalkoztak, GOLDEN négy
- elsősorban elméleti alapokon álló - megközelítést vázolt
[37] , amelyek:

- a GILMORE-GOMORY módszer,
- a 0-1 programozási feladat /EILON-CHRISTOFIDES/,
- szubgradiens optimalizáció /HELD-WOLFE-CROWDER/,
- "láda pakolás" /bin packing/ /JOHSON-DEMERS-ULLMAN-
-GAREY-GRAHAM/ .

Míg GOLDEN az algoritmusok felhasználására nem tér ki, addig HINXMAN csak a gyakorlati alkalmazások és eredmények bemutatását végzi el [56]. A feladatok elemzését a dimenziók szerint veszi, kihangsúlyozva a számítástechnikai háttér paramétereit /milyen a számítógép, az átlagos gépidő, a memória-igény, stb./.

A gyakorlatban előforduló feltételekről, hasznosságot befolyásoló tényezőkről STAINTON nyújt nagyon jó összefoglalót [102]. Nagyon hasznosnak bizonyult MADSEN munkája, amelyben az általa ismert publikációkat sorolja fel [85].

Összefoglalás

Dolgozatunkban az alapanyagok feldarabolásának számítógépes módszereit, a vonatkozó szakirodalmat tekintettük át, a gyakorlati alkalmazás szempontjai alapján kiemelve és részletezve három egymástól lényegesen eltérő körülmények között alkalmazható megközelítést. Részletesen ismertettük:

- a GILMORE-GOMORY módszert,
- HAESSLER heurisztikus módszerét,
- ADAMOWICZ-ALBANO-ORSINI heurisztikus eljárását.

Két alkalmazást is bemutattunk, az egyik az Orosházi Üvegyár részére készített siküveg-szabás /amely saját eredménynek tekinthető/, a másik a GILMORE-GOMORY módszer adaptálása Hewlett-Packard számítógépre, amelynek során a raktárkészlet-feltétel kezelésének egy lehetséges megoldását is elkészítettük. A számítógépes darabolások esetén a szállítási készség javítása, anyagmegtakarítás /csökken a hulladék/ és a kiszabási munka termékenységének javulása várható.

A dolgozat elsősorban a már megjelent, ill. megjelenés alatt levő saját publikációk /egyéni és közös/ felhasználásával készült el.

Hivatkozások

- [1] Adamowicz, M. - Albano, A.: A two stage solution of the cutting stock problem. in Information Processing 71 /Proc. IFIP congress '71/, Amsterdam: North-Holland /1972/, 1086-1091.
- [2] Adamowicz, M. - Albano, A.: A solution of the rectangular cutting-stock problem. IEEE Trans. on Sys. Man and Cyb. 6/1976/, 302-310.
- [3] Adamowicz, M. - Albano, A.: Nesting two-dimensional shapes in rectangular modules. Computer Aided Design 8/1976/, 27-33.
- [4] Albano, A.: A method to improve two-dimensional layout. Computer Aided Design 9/1977/, 48-52.
- [5] Albano, A. - Sapuppo, G.: Optimal allocation of two-dimensional irregular shapes using heuristic search methods. IEEE Trans. on Sys. Man and Cyb. 10/1980/ 242-248.
- [6] Albano, A. - Orsini, R.: Tree search approach to the m-partition and knapsack problems. Computer Journal 23/1980/, 256-261.
- [7] Albano, A. - Orsini, R.: A heuristic solution of the rectangular cutting stock problem. Computer Journal 23/1980/, 338-343.
- [8] Andrew, Chi-Chin Yao: New algorithm for bin packing. Journal of ACM 27/1980/, 207-227.

- [9] Barta, O. - Csuri, I.: Az acélszerkezet-gyártáshoz szükséges rud- és idomacélok anyagleszabási tervének elkészítése és programozása matematikai módszerek segítségével. Dunai Vasmű XX.évf./1980/, 57-63.
- [10] Biró, M.: Darabolási problémák megoldási módszerei. Előadás a X. Operációkutatási Konferencián, Debrecen /1980/.
- [11] Chambers, M.L. - Dyson, R.G.: The cutting stock problem in the flat glass industry - Selection of stock sizes. Operational Res. Quart. 27/1976/, 949-957.
- [12] Christofides, N. - Whitlock, C.: An algorithm for two-dimensional cutting problems. Operations Research 25/1977/, 30-44.
- [13] Christofides, N. - Mingozzi, A. - Toth, P.: Dynamic loading and unloading of liquids into tanks. Operations Research 28/1980/, 633-649.
- [14] Cloud, F.H.: Corrugator size and cutoff characteristics. TAPPI /Technical Association of Pulp and Paper Industry/ 62/1979/, 56-62.
- [15] Coffield, D.M. - Crisp, R.M.: Extensions to the coil slitting problem. Int. Journ. Prod. Res. 14/1976/, 625-630.
- [16] Cornwell, L.W. - Kalita, J.K.: The development of a computer program to automate the cutting of gun stock blanks. Model. Simul. 9/1980/, 1347-1352.

- [17] Coverdale, I. - Wharton, F.: An improved heuristic procedure for a nonlinear cutting stock problem. *Management Science* 22/1976/, 78-86.
- [18] Coverdale, I. - Wharton, F.: Cutting stock problems. *Journal of the Operat. Res. Soc.* 29/1978/, 503-504.
- [19] Cuthill, E. - Mckee, J.: Reducing the bandwidth of symmetric matrices. in. *Proceedings of 24th Nat. Conf. Assoc. Comput. Mach. ACM-pub. P-69, New York /1969/.*
- [20] Csirik, J. - Galambos, G. - Kovács, A. - Kuba, A. - Lengyel, I. - Papp, I.: Siküveg-szabás számítógépes optimalizálása az Orosházi Siküvegyárban. A Magyar Közgazdasági Társaság Operációkutatási Szakosztályának pályázatán nyertes pályamunka /1980/.
- [21] Csirik, J. - Galambos, G. - Kuba, A. - Lengyel, I.: Üvegszabás optimalizálása lineáris programozási modell segítségével. Előadás a X. Operációkutatási Konferencián, Debrecen /1980/.
- [22] De Cani, P.: A note on the two-dimensional rectangular cutting stock problem. *Journal of the Operat. Res. Soc.* 29/1978/, 703-706.
- [23] Dyson, R.G. - Gregory, A.S.: The cutting stock problem in the flat glass industry. *Operational Res. Quart.* 25/1974/, 41-53.
- [24] Eilon, S. - Christofides, N.: The loading problem. *Management Science* 17/1971/, 259-268.

- [25] Eisemann, K.: The trim problem. Management Science 3/1957/, 279-284.
- [26] Erdős, P. - Graham, R.L.: On packing squares with eures. Journ. of Comb. Theory /A/ 19/1975/, 119-123.
- [27] Farley, A.A. - Richardson, K.V.: Control of setups in cutting stock problems. Paper presented at EURO IV. Conference, Cambridge, England /1980/.
- [28] Filmer, P.J.: A sheet scheduling model for a digital computer. APPITA 24/1970/, 189-195.
- [29] Forgó, F.: Nemkonvex és diszkrét programozás. Közgazdasági és Jogi KK, 1978.
- [30] Forgó, F.: Matematikai programozás. Optimumszámítási modellek, 240-333.o., szerk.: Kósa A., 1979, Műszaki KK.
- [31] Garfinkel, R.S.: Minimizing wallpaper waste. Part I.: a class of travelling salesman problems. Operations Research 25/1977/, 741-751.
- [32] Gilmore, P.C.: Cutting stock, linear programming, knapsacking, dynamic programming and integer programming, some interconnections. Annals of Discrete Math. 4/1979/, 217-235.
- [33] Gilmore, P.C. - Gomory, R.E.: A linear programming approach to the cuttingstock problem. Operations Research 9/1961/, 849-859.

- [34] Gilmore, P.C. - Gomory, R.E.: A linear programming approach to the cutting stock problem. - Part II. Operations Research 11/1963/, 863-888.
- [35] Gilmore, P.C. - Gomory, R.E.: Multistage cutting stock problems of two and more dimensions. Operations Research 13/1965/, 94-120.
- [36] Gilmore, P.C. - Gomory, R.E.: The theory and computation of knapsack functions. Operations Research 14/1966/, 1045-1074.
- [37] Golden, B.L.: Approaches to the cutting stock problem. AIIE Transactions 8/1976/, 265-274.
- [38] Greenberg, H. - Hegerich, R.L.: A branch algorithm for the knapsack problem. Management Science 16/1970/, 327-332.
- [39] Greenberg, H. - Feldman, I.: Better step-pff algorithm for the knapsack problem. Discr. App. Math. 2/1980/, 21-25.
- [40] Haessler, R.W.: An heuristic programming solution to a nonlinear cutting stock problem. Management Science 17/1971/, B-793 - B-802.
- [41] Haessler, R.W.: Controlling cutting pattern changes in one-dimensional trim problems. Operations Research 23/1975/, 483-493.
- [42] Haessler, R.W.: Single-machine roll trim problems and solution procedures. TAPPI 59/1976/, 145-149.

- [43] Haessler, R.W.: A procedure for solving the film cutting-stock problem. ORSA-TIMS Meeting, Philadelphia /1976/.
- [44] Haessler, R.W.: A procedure for solving the 1.5-dimensional coil slitting problem. AIIE Transactions 10/1978/, 70-75.
- [45] Haessler, R.W.: Solving the two-stage cutting stock problem. OMEGA, The International Journal of Management Science 7/1979/, 145-151.
- [46] Haessler, R.W. - Vonderembse, M.A.: A procedure for solving the master slab problem in the steel industry. AIIE Transactions /11/1979/, 160-165.
- [47] Haessler, R.W. - Talbot, F.B.: A 0-1 model for solving the corrugator trim problem. Working paper No. 205, The University of Michigan /1980/.
- [48] Haessler, R.W.: Multimachine roll trim-problems and solutions. TAPPI 63/1980/, 71-74.
- [49] Haessler, R.W.: A note on computational modifications to the Gilmore-Gomory cutting stock algorithm. Operations Research 28/1980/, 1001-1004.
- [50] Hahn, S.G.: On the optimal cutting of defective sheets. Operations Research 16/1968/, 1100-1114.
- [51] Haim, M.J. - Freeman, H.: A multistage solution of the template-layout problem. IEEE Trans. on Sys. Science and Cybern. 6/1970/, 145-151.

- [52] Held, M. - Wolfe, P. - Crowder, M.: Validation of subgradient optimization. *Mathematical Programming* /6/1974/, 62-88.
- [53] Herz, J.C.: Recursive computational procedure for two-dimensional stock cutting. *IBM Journal of Res. and Dev.* 16/1972/, 462-469.
- [54] Hinxman, A.I.: Problem reduction and the two-dimensional trim-loss problem. *AISB Summer Conference on Artificial Intelligence and Simulation of Behaviour, University of Edinburgh* /1976/.
- [55] Hinxman, A.I.: A two-dimensional trim-loss problem with sequencing constraints. *Advance papers of IJCAI-'77. M.I.T.* /1977/, 859-864.
- [56] Hinxman, A.I.: The trim-loss and assortment problems: a survey. *European Journal of Operational Research* 5/1980/, 8-18.
- [57] Hóber, M. - Konkolyiné Soós, M.: Lemezszabástervek készítése számítógéppel heurisztikus eljárással. Előadás a X. Operációkutatási Konferencián, Debrecen /1980/.
- [58] Ingargiola, G.P. - Korsh, J.F.: A general algorithm for one-dimensional knapsack problems. *Operations Research* 25/1977/, 752-759.
- [59] Ironman, R.: Computerized corrugator scheduling: a way to improve efficiency. *Boxboard Containers* 10/1977/, 34-35.
- [60] Johnson, D. - Demers, A. - Ullman, J. - Garey, M. - Graham, R.: Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM Journal Comput.* 3/1975/, 297-325.

- [61] Johston, R.E. - Bourke, S.B.: The development of computer programs for rools deckle filling. APPITA 26/1973/, 444-448.
- [62] Johnston, R.E.: The cutting stock problem - Limiting the number of distinct cutting patterns. Technical report, Dept. of Math., University of Melbourne, Australia /1976/.
- [63] Johnston, R.E.: A new method for small cutting stock problems. Centre Technique du Papier, Grenoble, Oct. 1979 /C.R. No.1364/.
- [64] Johston, R.E.: Controlling cutting pattern changes on hard trim problems. Centre Technique du Papier, Grenoble, Oct. 1979 /C.R. No. 1365/.
- [65] Johnston, R.E.: Extensions to Haesslars heuristic for the trim problem. Centre Technique du Papier, Grenoble, Oct. 1979 /C.R. No. 1366/.
- [66] Johnston, R.E.: Linear programming applied to material balances. Centre Technique du Papier, Grenoble, Oct. 1979 /C.R. No. 1368/.
- [67] Johnston, R.E.: Cutting schedules for the paper and board industry. Proceedings of IFAC PRP Conference, Ghent, 1980.
- [68] Johnston, R.E.: Bounds for the one-dimensional cutting stock problem. Working paper, Footscray Inst. of Tech., Victoria, Australia /1980/.
- [69] Kalmár, J. - Lengyel, I.: Az alapanyagok feldarabolásának matematikai-számítástechnikai megoldása. Publikáció alatt.

- [70] Kantorovich, L.V.: Mathematical methods of organizing and planning production. Management Science 8/1962/, 366-422.
- [71] Komáromi, É.: Lineáris programozás. Optimumszámítási modellek, 119-239.o., szerk.: Kósa A., 1979, Műszaki KK.
- [72] Kovács, L.B.: A diszkrét programozás kombinatorikus módszerei. Bolyai J. Matematikai Társaság Operációkutatási tanfolyamának jegyzete, Budapest, 1969.
- [73] Kou, L.T. - Markowsky, G.: Multidimensional bin packing algorithms. IBM Journal of Res. and Dev. 21/1977/, 443-448.
- [74] Krakó, B.: Lineáris programozás. Közgazdasági és Jogi KK, Budapest, 1966.
- [75] KRUPP - ASCO Automatische Schnittplan - Optimierung. Prospektus.
- [76] Kuutti, M. - Voutilainen, R.: A discrete optimization approach to the pattern analysis of the cutting stock problem. Paper presented at EURO III. Conference, Amsterdam, Netherlands, 1979.
- [77] Lampl, T.: Anyagleszabási tervek készítése és programozása matematikai módszerekkel. Operációkutatási esettanulmányok /Számok/, 1971, 83-125.
- [78] Lengyel, I.: Siküveg szabásának optimalizálása. Diplomamunka, JATE /1979/.
- [79] Lengyel, I. - Kuba, A.: Siküveg szabásának optimalizálása /I/. SZIGMA 14/1981/, 169-190.

- [80] Lengyel, I.: A darabolási feladatok és a megoldásukra alkalmazott módszerek áttekintése. SZIGMA /közlés alatt/.
- [81] Litton, C.D.: A frequency approach to the one-dimensional cutting problem for carpet rolls. Operational Research Quart. 28/1977/, 927-938.
- [82] Lovász, L. - Gács, P.: Algoritmusok. /Matematika mőszakiaknak/ Mőszaki KK. 1978.
- [83] Madsen, Oli.B.G.: Glass cutting in a small firm. Mathematical Programming 17/1979/, 85-90.
- [84] Madsen, Oli.B.G.: A cutting sequencing algorithm. in: Optimization Techniques. edited by K. Iracki, K. Malanowski and S. Walukiewicz, Springer-Verlag 1980, 388-396.
- [85] Madsen, Oli.B.G.: References concerning the cutting stock problem. Working paper from IMSOR, Lyngby, 1980.
- [86] Madsen, Oli.B.G. - Kergard, K.E.: An application of travelling salesman routines to solve pattern allocation problems. Working paper from IMSOR, Lyngby, 1981.
- [87] Marconi, R.: Generalization of a mathematical procedure for the optimal solution of two-dimensional cutting problems. Technical reports No. 7., Centro Studi IBM of PISA, 1970.
- [88] Martello, S. - Toth, P.: Branch and bound algorithms for the solution of the general unidimensional knapsack problem. Paper presented at EURO II. Conference, Stockholm, 1976.

- [89] Martello, S. - Toth P.: A note on the Ingargiola-Korsh algorithm for onedimensional knapsack problems. Working paper, Istituto di Automatica, University of Bologna /1976/.
- [90] Martello, S. - Toth, P.: A computational study on large-size unidimensional knapsack problems. Presented TIMS/ORSA Joint National Meeting, San Francisco, 1977.
- [91] Niederhausen, H.P.: Programm zum automatischen Herstellen eines Schneidplans für beliebig gestaltete Werkstücke durch elektronische Datenverarbeitung. Schweissen und Schneiden 29/1977/, 103-105.
- [92] Nilsson, N.J.: Problem solving methods in artificial intelligence. McGraw-Hill, New York, 1971.
- [93] Page, E.: A note on a two-dimensional dynamic programming problem. Operational Research Quarterly 26/1975/, 321-324.
- [94] Pegels, C.C.: Heuristic scheduling models for variant of two-dimensional cutting stock problem. TAPPI 50/1967/, 532-535.
- [95] Pfefferkorn, C.E.: A heuristic problem solving design syste j for equipment or furniture layouts. Communications of the ACM 18/1975/, 286-297.
- [96] Philipson, R.H. - Ravindran, A.: Application of mathematical programming to metal cutting. Mathematical Programming Study 11/1979/, 116-134.

- [97] Reuter, S.: EDV-gestützte Erstellung der Fertigungunterlagen für den Materialzuschnitt. Fach. Blech Rohre Profile 27/1980/, 10.
- [98] Salkin, H.M. - De Kluyver, C.A.: The knapsack problem: a survey. Nav. Res. Log. Quart. 22/1975/, 127-144.
- [99] Sándor, G. - Lampl, T.: A nyirási probléma általánossága nagyméretű feladatoknál egy-, két- és háromdimenzió esetén. Előadás a X. Operációkutatási Konferencián, Debrecen, 1980.
- [100] Schaffer, P.: The small job-lot sheeter. TAPPI 61/1978/, 31-33.
- [101] Shapiro, S.D.: Performance of heuristic bin packing algorithms with remnants as random lengths. Inform. and Control 35/1977/, 146-158.
- [102] Stainton, R.S.: The cutting stock problem for the stockholder of steel reinforcement bars. Operational Res. Quarterly 28/1977/, 139-149.
- [103] Stainton, R.S.: Cutting stock problem - reply. Journal of the Operat. Res. Soc. 29/1978/, 505-506.
- [104] Tilanus, C.B. - Gerhardt, C.: An application of cutting stock in the steel industry. in: K.B. Haley /ed/, Operational Research '75 /Amsterdam, 1976/, 669-675.
- [105] Toth, P.: A new reduction algorithm for 0-1 knapsack problems. Presented at TIMS-ORSA National Meeting, Miami, 1976.
- [106] Toth, P.: Dynamic programming algorithm for the zero-one knapsack problems. Computing 25/1980/, 29-45.

- [107] Voutilainen, R.: On the pattern sequencing problem - definitions, complexity and applications. Presented at EURO IV. Conference, Cambridge, England, 1980.
- [108] Weingartner, H.M. - Ness, D.N.: Methods of solution of the multi-dimensional 0-1 knapsack problems. Operations Research 15/1967/, 83-103.
- [109] Wolfson, M.L.: Selecting the best lengths to stock. Operations Research 13/1965/, 570-585.

