GRÓSZ MIKLÓS

A hozzárendelési probléma egy általánosítása és annak megoldása

1. Bevezetés

Jelen dolgozatunkban az előregyártott elemekből megvalósított szerkezet-optimalizálási probléma matematikai modelljével és megoldásának módszerével foglalkozunk. A szerkezet-optimalizálási problémát úgy értelmezzük, hogy keressük az adott típuselem készletből előállítható, adott geometriájú szerkezet olyan tervét, melyben szereplő elemekre teljesülnek az egyensúlyi, kompatibilitási és lineáris korlátozó feltételek, továbbá valamilyen szempontból (súly, költség vagy ezek aránya) a szerkezet optimális.

Jelen dolgozatban nem foglalkozunk a szerkezet-optimalizálási probléma műszaki-gazdasági hátterével — ez egy másik dolgozat [17] tárgyát képezi — csak a probléma megoldását szolgáló algoritmus kidolgozásával.

A szerkezet-optimalizálási probléma matematikai megfogalmazásából kiindulva [17] a következő modellt kapjuk:

$$\sum_{i=1}^{n} c_i x_i \to \min, \tag{1}$$

$$\sum_{i \in Q_k} x_i = 1 \quad (k = 1, 2, \ldots, k^*); \quad \bigcup_{k=1}^{k^*} Q_k = I; \quad I = \{1, \ldots, n\}, \tag{2}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ki} x_i + b_{kij} x_i x_j) \geq s_k, \qquad (k = 1, 2, \ldots, k^*) \tag{3}$$

$$x_i \in \{0, 1\}, \qquad (i = 1, 2, \ldots, n)$$

ahol c_i — az i-edik elem súlya, költsége vagy ezek aránya
x_i = 1, ha az i-edik elemet alkalmaztuk és = 0 ellenkező esetben,
s_k, d_{ki}, b_{kij} — a probléma műszaki tartalmára jellemző paraméterek,
Q_k — a szerkezet k-adik helyén alkalmazható elem-típusok indexhalmaza.

Az (1)—(3) feladat abban különbözik az irodalomban ismertettektől (klasszikus hozzárendelési probléma, általánosított hozzárendelési probléma [10], speciális szállítási feladat [4]), hogy a (3) feltétel bal oldalán egy kvadratikus függvény áll, míg a fent említett problémákban vagy lineáris egyenlőtlenség állt, vagy egyáltalán nem szerepelt feltétel.

Ross és *Soland* [10] megmutatták, hogy a klasszikus hozzárendelési probléma a (4)—(6) feladatnak egy speciális esete:

$$\sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} \to \min, \tag{4}$$

$$\sum_{j\in J} \tau_{ij}x_{ij} \leq a_i, \qquad i\in I \tag{5}$$

$$\sum_{i\in I} x_{ij} = 1, \qquad j\in J \tag{6}$$

$$x_{ij}\in\{0,1\}, \quad i\in I, \quad j\in J.$$

Az (1)—(3) feladat esetén viszont belátható, hogy (4)—(6) ugyanakkor az (1)—(3) feladatnak egy speciális esete ($b_{kij}=0$).

2. A feladat transzformálása egy speciális „0—1" lineáris programozási feladatra

Felhasználva azt, hogy $x_i = x_i^2$, a (3) feltételeket átírhatjuk a következőképpen [14]:

$$\sum_{i+1}^{n}\sum_{j=1}^{n} a_{kij}x_ix_j \geq s_k \qquad (k=1,2,\ldots,k^*),$$

ahol

$$a_{kij} = \begin{cases} b_{kij}, & \text{ha } i\neq j \\ d_{ki}+b_{kii}, & \text{ha } i=j. \end{cases}$$

Vezessük be a következő jelöléseket:

$$A_k = \{a_{kij}\}$$

és

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_{kij}x_ix_j = X'A_kX,$$

ahol $X=(x_1,\ldots,x_n)$ és A_k $(n\times n)$-es mátrix. A továbbiakban mindig feltételezzük, hogy A_k szimmetrikus mátrix ($a_{kij}=a_{kji}$), mivel

$$X'A_kX = \frac{1}{2}X'(A_k+A_k')X,$$

ahol $\frac{1}{2}(A_k+A_k')$ már szimmetrikus mátrix.

Most bizonyítsuk be a következő tételt:

1. tétel. Az (1)—(3) feladatot a következő feladattá lehet transzformálni:

$$\sum_{i=1}^{n} c_i x_i \to \min, \tag{7}$$

$$\sum_{i\in Q_k} x_i = 1, \qquad (k=1,2,\ldots,k^*) \tag{8}$$

$$\sum_{\tau=1}^{k^*}\sum_{S=\tau+1}^{k^*}\sum_{i\in Q_\tau}\sum_{j\in Q_s}(a_{kii}x_i - 2a_{kij}z_{ij}) \geq s_k' \qquad (k=1,2,\ldots,k^*) \tag{9}$$

$$(P) \qquad x_i + x_j + z_{ij} \leq 2 \qquad \begin{pmatrix} i \in Q_\tau \\ j \in Q_s \\ \tau < s \\ \tau, s \in K \end{pmatrix} \tag{10}$$

$$x_i + z_{ij} \geq 1 \tag{11}$$

$$x_j + z_{ij} \geq 1 \tag{12}$$

$$x_i, z_{ij} \in \{0, 1\},$$

ahol

$$s'_k = s_k - \sum_{\tau=1}^{k^*} \sum_{s=\tau+1}^{k^*} \sum_{i \in Q_\tau} \sum_{j \in Q_s} 2a_{kij} \qquad (k = 1, 2, \ldots, k^*)$$

$$K = \{1, 2, \ldots, k^*\}.$$

Bizonyítás:

Vezessük be $\frac{n(n-1)}{2}$ új „0–1" változót és $3n(n-1)$ feltételt [5]. Ezek után az (1)–(3) feladat ekvivalens a következő feladattal, figyelembe véve, hogy A szimmetrikus mátrix:

$$\sum_{i=1}^{n} c_i x_i \to \min,$$

$$\sum_{i \in Q_k} x_i = 1, \qquad (k \in K)$$

$$\sum_{i=1}^{n} \sum_{j=i+1} (a_{kii} x_i + 2a_{kij} u_{ij}) \geq s_k \qquad (k \in K),$$

$$\begin{aligned} &x_i + x_j - u_{ij} \leq 1 \qquad \begin{pmatrix} i, j = 1, 2, \ldots, n \\ i < j \end{pmatrix} \\ &x_i \geq u_{ij} \\ &x_j \geq u_{ij} \\ &x_i, u_{ij} \in \{0, 1\} \qquad (i, j = 1, 2, \ldots, n). \end{aligned}$$

Az u_{ij} változó akkor és csak akkor egyenlő 1-el, ha $x_i = x_j = 1$, a többi esetekben $u_{ij} = 0$. Figyelembe véve a (8) feltételt, minden $Q_k (k \in K)$ halmazon egyidejűleg csak egy x_i változó lehet egyenlő 1-el, tehát

$$u_{ij} \equiv 0, \quad \text{ha} \quad j, i \in Q_k \quad (k \in K).$$

Ennek következtében az u_{ij} változók és a feltételek számát jelentősen lehet csökkenteni.

A továbbiakban feltételezhetjük, hogyha $k_1 < k_2$, akkor a Q_{k_1} halmaz tetszőleges eleme kisebb a Q_{k_2} tetszőleges eleménél, ellenkező esetben az ismeretlenek egyszerű átszámozásával ez mindig elérhető.

Ha áttérünk a $z_{ij} = 1 - u_{ij}$ „0–1" változóra, akkor ezzel bebizonyítottuk., hogy (1)–(3) feladat ekvivalens a (P) feladattal.

3. A speciális „0—1" lineáris programozási feladat megoldásának módszere

Ebben a fejezetben a (P) feladat megoldására bemutatunk egy olyan leszámlálási algoritmust, mely a *Geoffrion* és *Marsten* [7] által közölt, az egészértékű feladatok megoldására alkalmas általános algoritmus (general algorithmic framework) elvein alapszik. Ez az algoritmus három fő eljárásból tevődik össze: szeparálás (separation), relaxáció (relaxation) és kizárási kritérium (fathoming criterion).

3.1. Relaxáció

A javasolt algoritmus hatékonyságát nagyméretű (P) feladat esetén növeli az, hogy a nehezen kezelhető lineáris programozási feladat többszöri megoldása helyett felhasználjuk e feladat sajátosságát és relaxáció segítségével visszavezetjük a feladatunkat egy ritka mátrixszal rendelkező halmaz-lefedési probléma megoldására, melyre több hatékony algoritmus is ismert [1, 3, 8] az irodalomból.

A továbbiakban a következő jelöléseket fogjuk használni: $y = (x, z)$, a (8)—(10) feltételek együttható mátrixát jelöljük A-val, a (11)—(12) feltételek együttható mátrixát B-vel, és a megfelelő jobb oldalakat b-vel.

Most tekintsük meg a következő halmazlefedési problémát, melynél az együttható mátrix minden sorában két egyes áll:

$$cx + \lambda(b - Ay) \to \min,$$

$$\begin{aligned} x_i + z_{ij} &\geq 1 \quad \begin{pmatrix} i \in Q_\tau & j \in Q_s \\ \tau < s & \tau, s \in K \end{pmatrix} \\ x_j + z_{ij} &\geq 1 \\ x_i, z_{ij} &\in \{0, 1\} \quad (i, j = 1, 2, \ldots, n), \end{aligned}$$

ahol λ — Lagrange-féle multiplikátor.

$(\bar{P})$-vel jelöljük a (P) feladatnak megfelelő lineáris programozási feladatot, melyben y_i folytonos változó és teljesül a $0 \leq y_i \leq 1$ feltétel. A megfelelő célfüggvényértékeket $c(P)$, $c(\bar{P})$ és $c(PR_\lambda)$-val fogjuk jelölni, $\bar{\lambda}$-val pedig az $Ay \geq b$ feltételnek és a $(\bar{P})$ feladatnak megfelelő optimális Lagrange-féle multiplikátort.

Az, hogy a (PR_λ) feladat megoldása mennyire közelíti meg az eredeti (P) feladat megoldását, függ a λ Lagrange-féle multiplikátor választásától. Az ilyen értelemben legjobb λ vektort a következő feladat megoldásaként választhatjuk ki:

$$(D) \qquad \max_{\lambda \geq 0} c(PR_\lambda).$$

Itt meg kell jegyezni, hogy a (8) feltételek egyenlőségek, ezért a megfelelő λ_i értékekre nincs előjelkorlátozás. *Ross* és *Soland* [10] megmutatták, hogy ha a (8) feltételnek megfelelő λ_i értékeket egyenlővé tesszük a Q_i halmazon a második legkisebb együtthatóval, akkor a (P) feladat megoldása megegyezik a következő feladat megoldásával:

$$\sum_{i=1}^{n} c_i x_i - \sum_{k=1}^{k^*} \beta_k \left(1 - \sum_{i \in Q_k} x_i\right) \to \min,$$

$$\sum_{\tau=1}^{k^*} \sum_{s=\tau+1} \sum_{i \in Q_\tau} \sum_{j \in Q_s} (a_{kii} x_i - 2a_{kij} z_{ij}) \geq S'_k \quad (k = 1, 2, \ldots, k^*)$$

$$x_i + x_j + z_{ij} \leq 2 \qquad \begin{pmatrix} i \in Q_\tau & j \in Q_s \\ \tau < s & \tau, s \in K \end{pmatrix}$$

$$x_i + z_{ij} \geq 1$$

$$x_j + z_{ij} \geq 1$$

$$x_i \quad \text{és} \quad z_{ij} \in \{0, 1\} \qquad (i, j = 1, 2, \ldots, n),$$

ahol β_k — a második legkisebb c_i érték a Q_k halmazon. Tehát az általánosság megszorítása nélkül feltételezhetjük, hogy a D feladatban az egyenlőségnek megfelelő λ_i értékek konstansok.

A $(\bar{P})$, (PR_λ) és (D) feladatok kapcsolatára vonatkozóan *Geoffrion* [6] bebizonyította a következő tételt:

2. tétel. Ha a $(\bar{P})$ feladatnak létezik megoldása és a tetszőleges $\lambda \geq 0$ vektorra nézve teljesül

$$c(PR_\lambda) = c(\overline{PR}_\lambda), \tag{13}$$

akkor

$$c(\bar{P}) = c(PR_{\bar{\lambda}}) = c(D). \tag{14}$$

A (PR_λ) feladat B együttható mátrixa egy teljesen unimoduláris mátrix a következő tétel [18] alapján:

3. tétel. B mátrix teljesen unimoduláris, ha teljesülnek a következő feltételek:

1. Minden egyes eleme 0 vagy ± 1.
2. Egy oszlopban sincs több, mint két nullától különböző elem.
3. A B mátrix sorait két diszjunkt halmazra (R_1 és R_2) lehet bontani:

a) Ha egy oszlopban két hasonló előjelű, nullától különböző elem van, akkor az egyik R_1, a másik pedig R_2-nek az eleme;

b) ha egy oszlopban két különböző előjelű és nullától is különböző elem van, akkor mindkettő vagy az R_1-nek vagy az R_2-nek az eleme.

Figyelembe véve azt, hogy B teljesen unimodulárism átrix és a (PR_λ) feladat feltételeinek jobb oldalán egészértékű számok vannak, a (PR_λ) feladatra tetszőleges $\lambda \geq 0$ esetén teljesül (13), tehát a 2. tétel állítása is teljesül.

Ha a $(\bar{P})$ lineáris programozási feladat a méretei miatt vagy nagyon nehezen vagy egyáltalán nem oldható meg, akkor célszerű megoldani a (D) feladatot, mivel az az *Agmon-Motzkin-Shoenberg*-féle módszer segítségével (lásd 3.3 fejezet) visszavezethető egy sorozat halmazlefedési probléma megoldására különböző λ vektorokkal. Ha viszont a $(\bar{P})$ feladat megoldható, akkor a $\bar{\lambda}$ értékét nem a (D) feladat megoldásaként kapjuk, hanem a $(\bar{P})$ feladat megoldásának megfelelően állítjuk elő. A továbbiakban csak az első esettel foglalkozunk, mivel a szerkezet-optimalizálási feladat méretei általában nem teszik lehetővé a $(\bar{P})$ feladat megoldását.

Ha a $(PR_{\bar{\lambda}})$ feladat optimális $\bar{y}$ megoldására nem teljesül az $Ay \geq b$ feltétel, akkor szeparálási változók bevezetésével a megfelelő Q_k $(k \in K)$ halmazon (P^k) részfeladatokat kapunk. A szeparálási változók segítségével a megfelelő

Q_k ($k \in K$) halmazokat általában két diszjunkt részhalmazra bontjuk és egy speciális szeparálási feltételnek a (P) diszkrét programozási feladathoz való hozzáadásával kapjuk a (P^k) részfeladatokat.

3.2. Szeparálás

A szeparálási változó választásától nagymértékben függ az algoritmus hatékonysága, ezért ezeket mindig úgy szokták megválasztani, hogy minél jobban kihasználják a feladat sajátosságát.

Tegyük fel, hogy a Q_{k_1} halmazon a ($PR_{\bar{\lambda}}$) feladat optimális $\bar{y}$ megoldására nem teljesül az $Ay \geq b$ feltétel.

Valamilyen szempontból kiválasztjuk az $x_s, x_{s+1} \in Q_{k_1}$ szeparálási változókat, s akkor a (P^k) és a megfelelő ($PR^k_{\bar{\lambda}}$) feladatokat így írhatjuk fel:

$$(P^k) \qquad \left.\begin{matrix} c_1(P \mid x_\tau + x_{\tau+1} + \ldots + x_s = 0) \\ c_2(P \mid x_{s+1} + \ldots + x_t = 0) \end{matrix}\right\},$$

$$(PR^k_{\bar{\lambda}}) \qquad \left.\begin{matrix} c_1(PR_\lambda \mid x_\tau + \ldots + x_s = 0) \\ c_2(PR_\lambda \mid x_{s+1} + \ldots + x_t = 0) \end{matrix}\right\},$$

ahol

$$Q_{k_1} = \{\tau, \tau + 1, \ldots, s, s + 1, \ldots, t\}.$$

Ilyen szeparálási feltételeket először *Beale* és *Tomlin* [2] alkalmaztak olyan feladatok megoldásánál, ahol a (8) feltétel is szerepel.

Jelöljük K'-vel ($K' \subseteq K$) a Q_k halmazok azon indexeit, melyekre nem teljesül az $A\bar{y} \geq b$ Ha a (10) feltétel nem teljesül és $x_i \in Q_{k_1}$, valamint $x_j \in Q_{k_2}$, akkor $k_1, k_2 \in K'$. A megfelelő (P^k) ($k \in K'$) feladatok képezik a részfeladatok listáját (candidate list).

Az általánosság megszorítása nélkül a továbbiakban feltételezhetjük, hogy

$$c_i < c_{i+1} \quad (\forall i \in Q_k,\ k \in K),$$

ezek után térjünk vissza a szeparálási változók megválasztásával kapcsolatos szempontokra. Ezek a következők:

a) A ($PR_{\bar{\lambda}}$) feladat optimális $\bar{y}$ megoldására a Q_k halmazon nem teljesül a (8) feltétel, vagyis

$$\sum_{i \in Q_k} x_i^* > 1 \quad \text{vagy} \quad \sum_{i \in Q_k} x_i^* = 0.$$

Az első esetben a Q_k halmazt annyi páronként diszjunkt részhalmazra osztjuk szét, ahány 1-el egyenlő értékű ismeretlen van és azon belül a szeparálás úgy történjék, hogy minden részhalmazba csak egy ilyen ismeretlen kerüljön. A második esetben pedig, ha τ a legkisebb és t a legnagyobb index a Q_k halmazban, akkor a szeparálási változót a következőképpen választjuk:

$$s = \left[\frac{t - \tau}{2}\right].$$

b) A ($PR_{\bar{\lambda}}$) feladat optimális $\bar{y}$ megoldására a Q_k halmazon nem teljesül a (9) feltétel. Ha ugyanakkor a (8) feltétel sem teljesül, akkor a szeparálás az

a) pont szerint történik. Ellenkező esetben azt a változót, melynek értéke egyenlő 1-el, választjuk szeparálási változónak is.

c) A $(PR_{\bar{\lambda}})$ feladat optimális $\bar{y}$ megoldására a Q_{k_1} és Q_{k_2} halmazon nem teljesül a (10) feltétel. Ha ugyanakkor a (8) feltétel sem teljesül a Q_{k_1} vagy Q_{k_2} halmazon, akkor a szeparálás ezen a halmazon az a) pont szerint történik. Ha a (8) feltétel mindkét halmazon teljesül, de nem teljesül a Q_{k_1} vagy Q_{k_2} halmazon a (9) és (10) feltétel, vagy csak a (10) feltétel nem teljesül, akkor ezen a halmazon azt a változót választjuk szeparálási változónak is, melynek értéke egyenlő 1-el.

3.3 Kizárási kritérium

Az előző pontban leírtuk azt, hogy kapjuk meg a (P) feladatból kapott részfeladatok listáját. Ebben a pontban viszont leírjuk azt a kizárási kritériumot, amely szerint eldöntjük, hogy valamely részfeladatot figyelmen kívül hagyjunk-e a további vizsgálatokban vagy sem.

Ha a részfeladatok listája ismert, akkor megoldjuk a megfelelő $(PR_{\bar{\lambda}}^k)$ feladatokat. Jelöljük ezeket y^k-val. *Geoffrion* [6] megmutatta, hogy ha egy adott λ vektornak megfelelő y^k teljesíti a következő három feltételt:

1. y^k a $(PR_{\bar{\lambda}}^k)$ feladat optimális megoldása,
2. $Ay^k \geq b$,
3. $\lambda(b - Ay^k) = 0$,

akkor y^k a (P^k) feladat optimális megoldása.

Ha y^k a három feltételből csak az első két feltételt teljesíti, akkor y^k a (P) feladat ε-optimális megoldása, ahol

$$\varepsilon = \lambda(Ay^k - b).$$

Tehát a mi esetünkben, ha a $(PR_{\bar{\lambda}}^k)$ feladat optimális y^k megoldása nem teljesíti a 2. és 3. feltételt, vagyis y^k a (P^k) feladatnak nem megengedett megoldása, akkor módosítjuk a λ vektort. A 2. tétel alapján a λ vektor legjobb értékét a $(\bar{P}^k)$ vagy a (D^k) feladat optimális megoldásaként kapjuk. Nagyméretű (P) feladat esetén hatékonyabbnak látszik a (D^k) feladatok megoldása, mivel

a) a $(PR_{\bar{\lambda}}^k)$ feladat feltételeinek száma jelentősen kisebb, mint a megfelelő $(\bar{P}^k)$ lineáris programozási feladatnál;

b) a $(PR_{\bar{\lambda}}^k)$ feladat egy ritka mátrixszal rendelkező halmazlefedési probléma, melynek megoldására hatékony módszerek ismertek az irodalomból [1, 3, 8];

c) a $(PR_{\bar{\lambda}}^k)$ feladat struktúrája a (P) feladat megoldása folyamán változatlan marad (csak a szeparálási feltételek változnak), ami számítástechnikai szempontból nagyon előnyös.

Mint ahogy azt már említettük, a (D^k) feladat megoldására az *Agmon—Motzkin—Shoenberg*-féle módszert alkalmazzuk [15]. E módszer segítségével megoldunk egy sorozat $(PR_{\bar{\lambda}}^k)$ halmazlefedési feladatot, amelyben a λ_k^{v+1} vektor új értékét a következő képlet alapján határozzuk meg:

$$\lambda_k^{v+1} = \max\{\lambda_k^v + \theta_k^v(b - Ay_k^v), 0\} \qquad (v = 1, 2, \ldots),$$

ahol θ_k^v — egy pozitív szám, mely bizonyos feltételeknek felel meg [16],
λ_k^v — a (P^k) feladat jelenlegi Lagrange-féle multiplikátora.

A λ_k^1 $(k \in K')$ kiinduló értéket egy heurisztikus eljárás segítségével kapjuk, a megfelelő (P^k) feladat y_k^1 megengedett megoldása segítségével. A szerkezet-optimalizálási feladat műszaki tartalmából kiindulva konstruálható egy olyan heurisztikus eljárás, mely vagy a (P^k) feladat optimumához közelálló megengedett megoldást szolgáltat, vagy az a következtetés vonható le, hogy a (P^k) feladatnak nincs megoldása. Az utóbbi esetben a (P^k) feladatot töröljük a részfeladatok listájáról. Itt azonban erre nem térünk ki.

Amennyiben a $(PR^k[\lambda_k^\nu])$ feladatot egy leszámlálási módszer segítségével oldjuk meg, akkor az iterációk során mindig az előző feladat y_k^ν optimális megoldását használjuk fel a következő $(PR^k[\lambda_k^{\nu+1}])$ feladat korlátjaként, mivel y_k^ν a $(PR^k[\lambda_k^{\nu+1}])$ feladatnak is egy megengedett megoldása.

Ha valamely λ_k^ν vektornak megfelelő y_k^ν vektor teljesíti a 2. feltételt, vagyis y_k^ν a (P^k) feladat megengedett megoldása és $c(P_\nu^k) < z^*$, akkor a z^* korlát értéket egyenlővé tesszük a megfelelő $c(P_\nu^k)$ értékével. Ha $c(P_\nu^k) \geq z^*$, a megfelelő (P^k) feladatot kizárjuk a részfeladatok listájáról. Ugyanúgy kizárjuk a részfeladatok listájától ezt a feladatot, ha valamely y_k^ν vektorra teljesül a három feltétel, a z^* korlát értékét viszont ugyancsak egyenlővé tesszük a $c(P_\nu^k)$ értékével. Abban az esetben, ha a (PR_λ^k) feladatnak nincs megengedett megoldása, akkor nincs a megfelelő (P^k) feladatnak sem, mivel

$$H(P^k) \subseteq H(PR_\lambda^k),$$

ahol $H(P^k)$ és $H(PR_\lambda^k)$ megfelelően a (P^k) és (PR_λ^k) feladat lehetséges megoldásainak halmaza.

Ilyenkor a Q_k halmazon más szeparálási változót választunk és újra vizsgáljuk a módosított (P^k) feladatot.

Ha a részfeladatok listáját kimerítettük, akkor az aktuális korlátnak megfelelő megengedett megoldás egyúttal a (P) feladat optimális megoldása is. Ha viszont a részfeladatok megoldása során a (P) feladatnak egyetlen megengedett megoldása sem volt, akkor a (P) feladatnak nincs megoldása.

4. Az algoritmus leírása

1. Heurisztikus eljárás segítségével előállítjuk a (P) feladatnak egy y_1 megengedett megoldását és a megfelelő λ_1 vektort. A z^* korlát értékét egyenlővé tesszük a megfelelő $c(P)$ értékével. Ha a (P) feladatnak nincs megengedett megoldása, akkor a 11. lépésnél folytatjuk.
2. Megoldjuk a (D) feladatot, vagyis kiszámítjuk a $\bar{\lambda}$ vektor értékét.
3. Megoldjuk a $(PR_{\bar{\lambda}})$ feladatot. Ha $\bar{y}$ a $(PR_{\bar{\lambda}})$ feladat optimális megoldása és egyben a (P) feladatnak is, akkor a 11. lépésnél, egyébként a 4. lépésnél folytatjuk.
4. Előállítjuk a részfeladatok listáját.
5. A részfeladatokból kiválasztjuk azt, amelyre nézve teljesül:

$$\max_{i \in K'} \{\sum_j a_{ij} y_j^k - b_i\} \quad (y_0 = \bar{y}) \quad (k \in K').$$

Ha a részfeladatok listája üres $(K' = \emptyset)$, akkor a 10. lépésnél, egyébként a 6. lépésnél folytatjuk.

6. Heurisztikus eljárás segítségével kiszámítjuk a megfelelő λ_k^1 vektor értékét. Ha a (P^k) részfeladatnak nincs megoldása, akkor $K' = K'/\{k\}$ és visszatérünk az 5. lépésre, egyébként a 7. lépésnél folytatjuk.

7. Megoldjuk a (D^k) feladatot. A (D^k) feladat optimális megoldását jelöljük $\bar{\lambda}_k$-val.

8. Ha a $c(PR_{\bar{\lambda}_k}) \geq z^*$, akkor $c(P^k) \geq z^*$, tehát ezt a részfeladatot kihagyjuk a részfeladatok listájáról:

$$K' = K' \setminus \{k\}.$$

Visszatérünk az 5. lépésre, egyébként folytatjuk a 9. lépésnél.

9. Ha a $(PR_{\bar{\lambda}_k})$ feladat $\bar{y}^k$ optimális megoldása a (P^k) feladatnak egy megengedett megoldása, akkor $z^* = : c(P^k)$. Mindkét esetben a (P^k) részfeladatot kihagyjuk a listáról ($K' = K' \setminus \{k\}$ és visszatérünk az 5. lépésre.

10. Ha a részfeladatok listája üres és a (P) feladatnak találtunk legalább egy megengedett megoldást, akkor a z^* korlátnak megfelelő megengedett megoldás egyúttal optimális is.

11. Vége.

Az egyszerűség kedvéért az algoritmus 2. és 7. lépésben nem tértünk ki arra az esetre, ha a (D) vagy a (D^k) feladat megoldása során a $(PR[\lambda^i])$ vagy megfelelően a $(PR[\lambda_k^i])$ feladat optimális megoldása ugyanakkor a (P) feladatnak is egy megengedett megoldása.

Ebben az esetben, ha $c(PP[\lambda^i]) < z^*$ vagy $c(PR[\lambda_k^i]) < z^*$, akkor a korlát értékét megváltoztatjuk:

$$z^* = : c(PR[\lambda^i]) \quad \text{vagy} \quad z^* = : c(PR[\lambda_k^i]).$$

(Beérkezett: 1977. november 1-én.)

IRODALOM

1. Balas, E.: Set Covering with Cutting Planes from Conditional Bound. Report No. 399. Graduate School of Industrial Administration, Carnegie-Mellon University.
2. Beale, E. M.—Tomlin, J. A.: Special Facilities in a General Mathematical Programming System for Non-Convex Problems Using Ordered Sets. of the Fifth International Conference of Operations Research, Venice, Tavistock Publications, London, 1969.
3. Christofides, N.—Korman, S.: A Computational Survey of Methods for the Set Covering Problems. Management Science, Vol. 21. No. 5. 1975.
4. De Maio, A.—Roveda, C.: An All Zero-one Algorithm for a Certain Class of Transportation Problems. Oper. Res. 19(6) 1971. 1406—1408.
5. Forgó, F.: Egészszámú programozási feladatok néhány transzformációja. Szigma, VII. évf. 4. sz. 1974.
6. Geoffrion, A. M.: Lagrangean Relaxation for Integer Programming. Mathematical Programming Study 2 (1974), 82—114.
7. Geoffrion, A. M.—Marsten, R. E.: Integer Programming Algorithms: A Framework and State-of-the-art Survey. Management Science 18 (1972), 465—491.
8. Grósz, M.: Egy leszámlálási algoritmus a halmazlefedési probléma megoldására. Szigma, IX. évf. 1—2. sz. 1976.
9. Hammer, P. L.—Rudeanu, S.: Boolean Methods in Operations Research and Related Areas. Springer-Verlag, Berlin, 1968.
10. Ross, G. T.—Soland, R. M.: A Branch and Bound Algorithm for the Generalized Assignment Problem. Mathematical Programming. Vol. 8. (1975) No. 1.

11. TOMLIN, J. A.: Branch and Bound Methods for the Integer and Nonlinear Programming, North-Holland, Amsterdam, 1970.
12. LAWLER, E. L.: The Quadratic Assignment Program: A Brief Review. In Roy B. (ed.): Combinatorial Programming: Methods and Applications. Reidel Publishing Company, Dordrecht-Holland, 1975.
13. HANAN, M.—KURTZBERG, J. M.: A Review of the Placement and Quadratic Assignment Problems. SIAM Rev. 14 (1972).
14. HAMMER, P.—RUBIN, A.: Some Remarks on Quadratic Programming with 0—1 Variables. Revue Française d'Informatique et de Recherche Operationelle, 4., 1970.
15. HELD, M.—KARP, R.: Travelling Salesman Problem and Minimum Spanning Trees. Mathematical Programming. 1. (1971) 6—26.
16. HELD, M.—WOLFE, P.—CROWDER, H.: Validation of Subgradient Optimization. Mathematical Programming, Vol. 6. (1974) No. 1.
17. GRÓSZ, M.: Automatizált tervezés integer programozással. Műszaki Tudomány, 53. 1977. 207—216.
18. GARFINKEL, R. S.—NEMHAUSER, G. L.: Integer Programming. New York, 1972. John Wiley and Sons.

THE GENERALIZATION OF THE ASSIGNMENT PROBLEM AND ITS SOLUTION

We deal with a generalization of the assignment problem obtained from the mathematical model of a structure optimization problem. Furthermore, a theorem will be proved whereby our problem is reduced to the solution of a "0—1" linear programming problem with variables of n^2 order of magnitude. Making use of the particularity of the problem an enumeration algorithm is presented consisting of the solution of a series of set covering problems.

ОДНО ИЗ ОБОБЩЕНИЙ ЗАДАЧИ О НАЗНАЧЕНИИ И ЕЕ РЕШЕНИИ

В данной статье изучается одно из обобщений задачи о назначении, которая получается в результате записи математической модели задачи оптимизации конструкции здания. Доказывается, далее, теорема, посредством которой наша задача сводится к решению целочисленной задачи «0—1» программирования с числом переменных порядка n^2. Используя особенности данной задачи приводится такой алгоритм перебора, который основывается на многократном решении задачи о покрытии.