Ütemezési modellek

Az ütemezési problémák osztályozása

Az ütemezési problémákban adott m darab gép és n számú munka, amelyeket az $1, \ldots, n$ számokkal fogunk sorszámozni. A feladat az, hogy ütemezzük az egyes munkák végrehajtását a gépeken úgy, hogy valamely cél szerint optimális ütemezést kapjunk. A legtöbb modellben a *munkák végrehajtásának ütemezésén* vagy egyszerűbben a *munkák ütemezésén* azt értjük, hogy a j-edik munkát hozzárendeljük valamely géphez egy S_j *kezdési* és C_j *befejezési idővel* minden j-re. A munkákhoz tartozik egy *végrehajtási idő*, amit p_j-vel szokás jelölni. Ez adja meg, hogy mennyi ideig tart a munkát elvégezni. Ennek megfelelően a munkához rendelt kezdési és befejezési időkre a $C_j - S_j = p_j$ feltételnek kell teljesülni.

A különböző modelleket többféle szempontból is osztályozhatjuk. A továbbiakban a legismertebb változatokat igyekszünk összegyűjteni.

A munkák meghatározó paraméterei

Mint már említettük a munkákhoz tartozik egy végrehajtási idő, de más modellekben egyéb paraméterei is vannak az egyes munkáknak.

- A munkákhoz rendelhető *érkezési idő* is, ezt a paramétert a j-edik munkára általában r_j jelöli. Ez az idő azt az időpontot adja meg, amelytől kezdve a munka végrehajtása elkezdhető, tehát a munkára az $S_j \geq r_j$ feltételnek kell teljesülni.

- A j-edik munkához tartozhat egy d_j *határidő*. Itt két különböző típusú modellt vizsgálhatunk. Az első esetben csak olyan ütemezéseket fogadunk el, amelyekre $C_j \leq d_j$, azaz amelyek betartják a határidőt, a másodikban megszeghetjük a határidőt, de ekkor a célfüggvényben a határidő is szerepel.

- A j-edik munkához hozzárendelhetünk egy w_j *súlyt* vagy egy $f_j(t)$ *súlyfüggvényt*, amely azt adja meg, hogy mennyire fontos a munka, illetve azt, hogy mennyire fontos a munka t időpontig való befejezése.

Egy másik fontos osztályozási szempont az, hogy mi a függvény, aminek az optimumát keressük. Eszerint a szempont szerint is igen sok modell került bevezetésre, az alábbiakban a legfontosabbakat vázoljuk. A célfüggvények két alapvető osztályra bonthatók: a maximum célfüggvényekre és az összeg célfüggvényekre.

- Talán a legismertebb modellek azok, amelyekben az utolsónak befejezett munka befejezési idejét akarjuk minimalizálni, azaz ahol a célfüggvény $\max\{C_j : 1 \leq j \leq n\}$. Szintén gyakran használt célfüggvény a *teljes befejezési idő*, amely a befejezési idők összege. Általánosabb esetben, mikor a munkáknak van súlya vagy súlyfüggvénye, akkor a $\sum_{j=1}^{n} w_j \cdot C_j$ illetve $\sum_{j=1}^{n} f_j(C_j)$ függvényeket minimalizálhatjuk.

- Amennyiben a munkákhoz határidő is tartozik, akkor a célfüggvény általában a késések minimalizálása. Itt két értéket szokás vizsgálni. Az első a *késési idő* (lateness), amely az $L_j = C_j - d_j$ érték, a másik pedig a *csúszási idő* (tardiness), amely a $T_j = \max\{0, L_j\}$ érték. A két maximum célfüggvény a késési időknek illetve a csúszási időknek a maximuma. Egyéb modellekben ezen értékek összegét illetve súlyozott összegét igyekszünk minimalizálni. Itt érdemes azt a modellt említenünk, ahol a cél az elkésett ($T_j > 0$) munkák számának minimalizálása.

- Amennyiben érkezési idők is vannak szokás a befejezési idő helyett a *folyási időt* (flow time) vizsgálni, amely az $F_j = C_j - r_j$ érték. Ezekben a modellekben a célfüggvény ezen F_j értékek maximuma vagy súlyozott összege.

A fenti alapvető osztályozáson kívüli egyéb változatokat, általánosításokat kaphatunk néhány extra feltétellel. Az alábbiakban ezekből gyűjtöttünk össze néhányat.

- Feltehetjük, hogy bizonyos munkák megkövetelik, hogy egyéb munkák már végre legyenek hajtva. Igen sok gyakorlati problémánál előfordulnak ilyen feltételek. Ekkor az egyes munkákhoz tartozik az a feltétel is, hogy mely munkák előzetes végrehajtását követelik meg. Ezeket a feltételeket egy irányított gráffal írhatjuk le, amelyet precedencia gráfnak hívunk. Ezen extra kiegészítés mellett az összes, a fentiekben említett modell vizsgálható.

- Az eddigiek során végig azzal a feltétellel éltünk, hogy minden egyes munkához pontosan egy végrehajtási idő tartozik és ez független attól, hogy melyik gépen kerül a munka végrehajtásra. Ez általában gyakorlati problémáknál nem így van. Egy általánosabb modellben minden munkához egy *végrehajtási vektor* tartozik, amely i-edik komponense megadja, hogy az M_i gépen mennyi ideig tart a munkát végrehajtani. Amennyiben a vektor tetszőleges lehet, független gépekről beszélünk. Itt két további speciális esetre kell kitérnünk. Az egyik esetben a j-edik munkához egy W_j végrehajtási súly tartozik. A végrehajtási vektorának i-edik komponense W_j/v_i, ahol v_i az M_i gép sebességének felel meg. Ezt az esetet összefüggő gépek esetének nevezzük. A második esetben a korlátozott hozzárendelési esetben a végrehajtási vektor néhány komponense végtelen, a többi megegyezik. Ez azt modellezi, hogy a gépek azonosak csak a munka néhány gépen nem hajtható végre.

- Egy további modell, amelyben megengedjük, hogy a munkák végrehajtása megszakítható legyen. Ekkor a j-edik munkához nem egy darab legalább p_j hosszú intervallumot kell hozzárendelnünk valamely gépen, hanem több, egymást nem átfedő intervallumot (akár különböző gépeken), amelyek összhossza p_j.

A fentiekben csak olyan modelleket ismertettünk, amelyekben a munkák pusztán egyetlen részből álltak. Számos gyakorlati probléma esetén egy munka több műveletből állhat. Az ilyen problémák egy nagy csoportját leíró modelleket shop ütemezésnek nevezzük. Itt csak azokat a modelleket tekintjük át, amelyekben minden egyes művelet egy, a művelethez rendelt gépen hajtható végre, és adott a művelet végrehajtási ideje. Ezen osztályon belül két fő csoportot különböztetünk meg. Amennyiben bármely munkára a hozzátartozó műveleteket tetszőleges sorrendben végrehajthatók, akkor *open shop ütemezésről* beszélünk. A másik esetben, amelyben a munkákhoz tartozó műveleteket az adott sorrendben kell végrehajtanunk ismét megkülönböztetünk két modellt. Ha a munkák műveleteihez tartozó gépek sorrendje nem azonos akkor *job shop ütemezésről* beszélünk. Ha a gépek sorrendje azonos minden munkára, akkor *flow shop ütemezés* modellt kapjuk.

A fentieken kívül még több változat létezik, a jegyzet célja nem az egyes modellek részletes összegyűjtése. Az egyes modellek osztályozása során kidolgoztak egy három mezőből álló jelölésrendszert. Az első mező írja le a probléma struktúráját (pl. hány gép van, azonosak -e, shop problémáról van -e szó), a második mező a munkák tulajdonságait írja le (pl. érkezési idők, határidők, precedencia feltételek), a harmadik mező a célfüggvényt adja meg. Ezt a jelölésrendszert itt nem tárgyaljuk tovább. Példákat mutatunk a későbbiekben, amikor használjuk ezt a jelölésrendszert, de mindig kifejtjük részletesebben is milyen problémáról van szó.

A különböző változatok részletesebb osztályozása, és az egyes modelleket leíró három részből álló jelölésrendszer megtalálható a szakirodalomban, például a [3] dolgozatban.

Felírás matematikai programozási feladatként

Több ütemezési probléma felírható matematikai programozási feladatként (lineáris programozási feladat, egészértékű programozási feladat), és így az ezen általános problémák megoldására kidolgozott eljárások használhatóak az adott ütemezési feladat megoldására is. Ebben a részben néhány ilyen példát mutatunk be.

Lineáris programozási feladat

Az operációkutatás egyik legismertebb problémája a lineáris programozási feladat. Lineáris programozási feladatról beszélünk olyan optimalizálási feladatok esetén, amikor egy lineáris függvény szélsőértékét keressük lineáris feltételek mellett. A lineáris programozási feladat megoldására a legismertebb eljárás a szimplex módszer (ld. [1]), amely annak ellenére, hogy elméleti szempontból exponenciális időigényű algoritmus a gyakorlatban igen hatékonynak bizonyult. Fontosnak tartjuk megjegyezni, hogy ismert polinomiális időigényű algoritmus is a lineáris programozási feladat megoldására.

Egyes ütemezési problémák felírhatók lineáris programozási feladatként. Példaként a $Pm|prmp|C_{max}$ problémát tekintjük, (m párhuzamos azonos gépen ütemezzük a munkákat a maximális befejezési időt minimalizálva, a munkák megszakítását engedélyezve).

$Pm|prmp|C_{max}$ **lineáris programozási feladatként**

Jelölje az $x_{ij} \geq 0$ változó, azt az időmennyiséget, amelyet az i-edik gép összesen a j-edik munkával töltött. Ekkor a feladat a következő formában írható fel:

$$\min C_{max}$$

$\sum_{i=1}^{m} x_{ij} = p_j, \quad j = 1, \ldots, n$

$\sum_{i=1}^{m} x_{ij} \leq C_{max}, \quad j = 1, \ldots, n$
$\sum_{j=1}^{n} x_{ij} \leq C_{max}, \quad i = 1, \ldots, m,$
$x_{ij} \geq 0, \quad i = 1, \ldots, m, \; j = 1, \ldots, n.$

Ekkor az első csoportja a feltételeknek azt adja meg, hogy összesen a gépek a j-edik munkával valóban p_j időt töltenek. A második feltételcsoport azt biztosítja, hogy a teljes idő amelyet egy munkán a gépek összesen dolgoznak legfeljebb a maximális befejezési idő (ezeket a feltételeket helyettesíthetjük a $p_j \leq C_{max}$ feltételekkel). A harmadik csoportja a feltételeknek azt biztosítja, hogy egyetlen gép sem dolgozik többet, mint C_{max}.

Egészértékű programozási feladat

Az egészértékű programozási feladat olyan lineáris programozási feladat, amelyben a változókról kikötjük, hogy egészek. Az egészértékű programozási feladat megoldására kidolgozott eljárások legtöbbje a vágósíkok módszerén vagy a korlátozás és szétválasztás módszerén alapul. Ezen módszerek részletes tárgyalása az egészértékű programozási feladat megoldására megtalálható a [1] és [2] egyetemi jegyzetekben.

Sok ütemezési probléma megadható egészértékű programozási feladatként. Példaként az $1||\sum w_j C_j$ problémát tekintjük azt a problémát, amelyben egyetlen gép van, a munkákat két paraméter határozza meg a végrehajtási idő és a munka súlya, és amelyben a cél minimalizálni a teljes súlyozott befejezési időt ($\sum w_j C_j$). Két különböző reprezentációt is megadunk, mindkét módszer számos ütemezési probléma esetén használható egészértékű programozási feladat megkonstruálására. Érdemes megemlíteni, hogy a probléma megoldható egy egyszerűbb algoritmussal (amit később ismertetünk) az alábbi interpretáció felhasználása nélkül is.

$1||\sum w_j C_j$ **egészértékű programozási feladatként I.**

Legyen x_{jk} egy döntési változó, amely 1 ha a j munka az ütemezésben hamarabb kerül végrehajtásra, mint k és 0 egyébként. Ezen változók segítségével az alábbi programozási feladat írja le az ütemezési problémát.

$$\min \sum_{j=1}^{n} \sum_{k=1}^{n} w_j p_k x_{kj} + \sum_{j=1}^{n} w_j p_j$$

$$\begin{array}{ll} x_{kj} + x_{jk} = 1, & j, k = 1, \ldots, n, j \neq k \\ x_{kj} + x_{jl} + x_{lk} \leq 2 & j, k, l = 1, \ldots, n, j \neq k, k \neq l, l \neq j \\ x_{jk} \in \{0, 1\} & j, k = 1, \ldots, n \end{array}$$

$x_{jj} = 0 \qquad\qquad\qquad\qquad j = 1, \ldots, n.$

Mivel a j-edik munka befejezési ideje $\sum_{k=1}^{n} p_k x_{kj} + p_j$, ezért a célfüggvény valóban $1||\sum w_j C_j$, és az is könnyen ellenőrizhető, hogy a többi feltétel pontosan azt írja le, hogy az x_{jk} változók a munkák egy sorbarendezését adják meg. Érdemes megjegyezni, hogy a fenti programozási feladat könnyen kiterjeszthető arra az esetre, amelyben a munkákra precedencia feltételek vannak előírva, minden ilyen feltétel rögzíti valamely x_{jk} változó értékét.

A fenti típusú interpretáció, ahol döntési változók a munkák sorrendjét adják meg, nem terjeszthető ki egynél több gép esetére. Az alábbiakban egy olyan egészértékű programozási interpretációt adunk meg, amely kiterjeszthető több gép esetére. Ezen interpretáció használatához fel kell tennünk, hogy a végrehajtási idők egészek. Könnyen adódik, hogy ekkor elegendő azokat a lehetséges ütemezéseket vizsgálni, amelyekben a kezdési idők is egészek. Továbbá fontos megjegyeznünk, hogy az alábbi egészértékű programozási feladatban a változók száma rendkívül nagy lehet.

$1||\sum w_j C_j$ **egészértékű programozási feladatként II.**

Legyen az x_{jt} változó 1 ha a j-edik munka a t egész időpontban kezdődik, 0 különben. Legyen $l = \sum_{j=1}^{n} p_j - 1$. Ekkor az alábbi programozási feladat írja le az ütemezési problémát:

$$\min \sum_{j=1}^{n} \sum_{t=1}^{l} w_j (t + p_j) x_{jt}$$

$\sum_{t=0}^{l} x_{jt} = 1, \qquad j = 1, \ldots, n$
$\sum_{j=1}^{n} \sum_{s=\max\{t-p_j,0\}}^{t-1} x_{js} = 1, \quad t = 0, 1, \ldots, l,$
$x_{jt} \in \{0, 1\}, \qquad j = 1, \ldots, n,\ t = 0, \ldots, l.$

A változók fentiekben megadott értelmezése mellett egyből adódik, hogy a célfüggvény valóban a teljes súlyozott befejezési idő. Az első feltételcsoport azt biztosítja, hogy minden munkára pontosan egy kezdési időt határozunk meg. A második feltételcsoport pedig azt biztosítja, hogy minden időpontban csak egy munkát hajtunk végre a gépen.

Hivatkozások

[1] Imreh B., Operációkutatás, JATEPress, Szeged, 1994.

[2] Imreh B., Kominatorikus Optimalizálás, NOVODAT, 1999.

[3] B. Chen, C. N. Potts, G. J. Woeginger, A review of machine scheduling, in *Handbook of Combinatorial Optimization, Volume 3.* eds. D. Z. Du, P. Pardalos, Kluwer Academic Publisher, 1998, 21–170.