Production Systems and Information Engineering
Volume 10 (4), pp. 11–23.
http://doi.org/10.32968/psaie.2022.4.2

HATÁRIDŐS TEVÉKENYSÉGEK VÉGREHAJTÁSÁNAK ÜTEMEZÉSE IDŐBEN VÁLTOZÓ RENDELKEZÉSRE ÁLLÁSÚ PÁRHUZAMOS ERŐFORRÁSOK ESETÉBEN

KULCSÁRNÉ FORRAI MÓNIKA
Miskolci Egyetem, Informatikai Intézet
aitkfm@uni-miskolc.hu

KULCSÁR GYULA
Miskolci Egyetem, Informatikai Intézet
iitkgy@uni-miskolc.hu

Absztrakt. Napjainkban a gyártásütemezési feladatok modellalapú megoldásai egyre nagyobb szerepet kapnak a kézzel készített, szabályalapú táblázatkezelős megoldásokkal szemben. Ebben a cikkben bemutatunk egy olyan modellezési megközelítést, mely előnyösen alkalmazható olyan valós ütemezési feladatok megoldására, ahol időben változó rendelkezésre állású párhuzamosan működő erőforrásokat használnak, és a munkák egy műveletből állnak, valamint indítási időkorláttal és befejezési határidővel is rendelkeznek. Az ütemezés (optimalizálás) célja a legnagyobb késés minimalizálása. A cikkünk bemutat egy időtartalék-orientált algoritmust, mely a vizsgált ütemezési feladat optimális megoldását polinomiális futási idő alatt állítja elő.

Kulcsszavak: ütemezés, modellezés, optimalizálás, algoritmus

1. Bevezetés

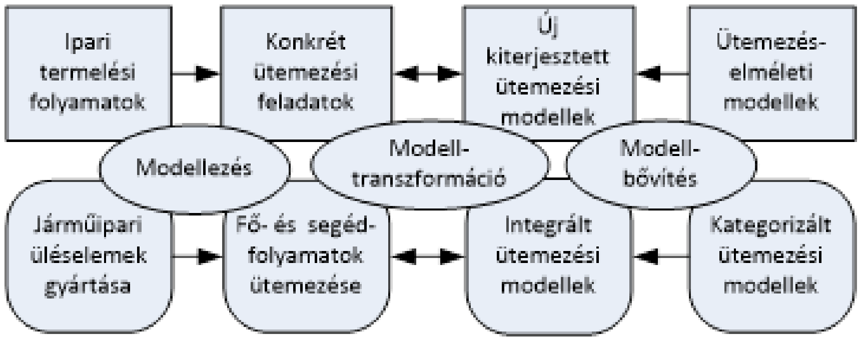
A termelővállalatoknak és termelési hálózatoknak egyre fokozódó kihívásokkal kell szembenézniük. A globalizált piaci környezetben a versenyképesség növelése érdekében nagyon fontos, hogy a termelőrendszerek gyorsan alkalmazkodjanak az üzleti, piaci, technológiai és informatikai változásokhoz. Ennek érdekében a termelési folyamatok hatékonyságát és rugalmasságát növelni kell, valamint a szállítókészséget folyamatosan javítani kell, miközben a készletszinteket, a termelési költségeket és a határidő-túllépéseket alacsony értéken kell tartani.

Ezeket a feladatokat a termelésmenedzsment operatív szintű funkciói oldják meg. A korszerű vállalatirányítási tevékenység kiterjed a vállalat működésével összefüggő összes területre. A modern termelési, menedzsment és informatikai paradigmákra alapozva egy vállalat egészének működését és annak folyamatait a tervezés, az előkészítés, a végrehajtás és az ellenőrzés szakaszaiból felépített többszörösen visszacsatolt szabályzórendszerek irányítják. Az ütemezés egyre fontosabb szerepet játszik a gyártórendszerek hatékony irányításában.

Az időben változó rendelkezésre állású erőforrásokkal kapcsolatos ütemezési téma kutatásának motivációját egy járműipari gyártórendszerben felmerülő valós ütemezési feladat problémáinak megoldása adta. A konkrét ipari feladat vizsgálata során egyértelművé vált, hogy vannak olyan ütemezési feladatok, amelyeknél a termelési finomprogramozás megfelelő minőségű elvégzéséhez a gyártási főfolyamat mellett a gyártáselőkészítési mellékfolyamat ütemezése is feltétlenül szükséges. A főfolyamatok és az előkészítő mellékfolyamatok ütemezése egymástól kölcsönösen függő feladatok megoldását követeli meg.

2. A modellezés szerepe ipari ütemezési feladatok megoldásában

A kutatás alapvető célja az volt, hogy kidolgozzuk az ütemezési probléma részletes modelljét és annak megoldási algoritmusát annak érdekében, hogy ezek alapján implementálható legyen egy olyan ütemező szoftver, amely el tudja készíteni a mindenkori aktuális határidős tevékenységek végrehajtási ütemtervét úgy, hogy a legnagyobb határidő-túllépés (csúszás) minimális legyen.



1. ábra. A modellezés szerepe az ipari ütemezési feladatok megoldásában

Az ipari ütemezési feladat megoldása érdekében két irányból közelítettük meg a problémát (1. ábra). Az egyik irány a vizsgált ipari folyamat megismerésével kezdődik. A folyamatok vizsgálata során alapvetően az ütemezési alapegységek, a műveletek, a technológiai lépések/fázisok/útvonalak, a végrehajtási jellemzők, az erőforrások és képességeik, a lehetséges alternatívák és korlátozások jellemzőit kell összegyűjteni és rendszerezni. Ezek után kerülhet sor a konkrét ütemezési feladatok modellezésére. Definiálni kell az erőforrás-környezetet, amely az erőforrásokat, az operációkat és azok kapcsolatait foglalja magában. Pontosan meg kell határozni az ütemezési alapegység fogalmát, valamint a végrehajtási jellemzőit és korlátozásait. Ezek után a termelésmenedzsment és a gyártásirányítás elvárásait és követelményeit ütemezési célok formájában kell kifejezni a hozzájuk tartozó fontossági mutatókkal (prioritásokkal) együtt. Mindezek együtt definiálják a konkrét ütemezési feladat részletes modelljét.

A másik irány a szakirodalomban elérhető operációkutatási és ütemezés-elméleti modellek vizsgálatával és azok alkalmazási feltételeinek elemzésével kezdődik. A szakirodalomban nagyon sokféle ütemezési modell található. Tematikusan szerkesztett könyvek (pl.: [4], [12], [13]) és különböző témakörökre koncentráló összefoglaló szakcikkek (pl.: [1], [2], [9]) segítik a tájékozódást. Erőforrás-környezeti nézőpontból a vizsgált ipari feladathoz a párhuzamos erőforrások modelljei (Parallel Machines) állnak a legközelebb. Ennek a témakörnek a részleteit jól összefoglalják pl. az [5], [7], és [11] cikkek. Az optimalizálási célok szempontjából a modellek döntő többsége egyetlen optimalizálási célfüggvénnyel foglalkozik. A direkt megrendelésre gyártás esetében főként valamilyen határidő-orientált teljesítménymutató kerül a középpontba. Az ütemezési modellek csak egy szűkebb halmazára jellemző a gyakorlati igények szempontjából fontos több-kritériumos (multi-objective) szemlélet (pl.: [3], [8], [9], [10], [14]).

Az operációkutatási és matematikai szemlélettel megalkotott elméleti ütemezési modellekben a konkrét gépek, a dolgozók és egyéb erőforrások műveletvégző képességei, időben változó rendelkezésre állásai, a valós folyamatokhoz kapcsolódó egyéb folyamatok (pl.: logisztikai műveletek, gyártástechnológiai előkészítő tevékenységek stb.) gyakran túlzottan le vannak egyszerűsítve, vagy teljesen figyelmen kívül vannak hagyva. Ezek a modellek kiindulási alapként felhasználhatók, de a konkrét ipari problémák igényei miatt jelentős továbbfejlesztésre és kiterjesztésre van szükség.

A kiterjesztett modellek és a valós problémák összekapcsolásának egyik lehetséges módszere a problématér transzformációjára alapozott szemantikai megfeleltetés. Ennek lényege, hogy a valós feladat tényleges döntési változóinak halmazát (vagy annak egy részhalmazát) a kapcsolódó korlátozásokkal együtt egy célszerűen megkonstruált (pl. szimulációs) eljárás visszavezeti egyszerűbben kezelhető döntési változók halmazára és azok korlátozásaira. Ezt a transzformált problémát a hozzá pontosan illeszkedő kiterjesztett modell oldja meg és az így kapott eredményből inverz transzformációval kiadódik az eredeti feladat megoldása.

A cikk további részében ennek a kétirányú szemléletnek az alkalmazását mutatjuk be egy példaként választott ütemezési probléma megoldásán keresztül.

3. A vizsgált ütemezési feladat modellezése

3.1. Az ütemezési feladat jellemzői

Adott egy gyártórendszer, amely járműipari alkatrészeket gyárt. A gyártási főfolyamat megfelelően előkészített készülékkonfigurációkat is igényel. Egy készülékkonfiguráció tartalmaz egy formahordozót és egy vagy több formát. A konfigurációkat szerelő szakmunkások állítják össze. A dolgozók időbeli rendelkezésre állása nem folytonos, hanem előzetes műszakbeosztáshoz kötött.

A vizsgált ütemezési feladat a következőképpen foglalható össze:

- adott számú egymástól független munkát (tevékenységet) kell elvégezni;
- minden egyes munkának egyedileg definiált legkorábbi indítási időpontkorlátja és legkésőbbi befejezési határideje van;
- minden egyes munkához egyetlen operáció tartozik;
- az operációk műveleti ideje adott;
- az operációk (műveletek) nem szakíthatók meg;
- az erőforrások rendelkezésre állása nem folyamatos. Adott időintervallumokban (műszakokban) végezhetők el a műveletek;
- a műszakok rögzített hosszúságúak (időtartamúak) és diszjunktak (nem lapolódhatnak át);
- a munkákat szerelő szakmunkások végzik el;
- az egyes műszakokban rendelkezésre álló dolgozó szerelők száma adott;
- a dolgozók műveletvégzési sebessége ismert;
- az ütemezés célja az, hogy a korlátozások betartásával minden munka sikeresen befejeződjön a saját határidejére.

Ez az ütemezési feladat ebben az általános formában nagyon nehezen kezelhető. Ezért kidolgoztunk egy probléma-transzformációs eljárást, melynek segítségével a problémát átalakítjuk egy továbbfejlesztett, párhuzamosan működő absztrakt gépeket (erőforrásokat) tartalmazó ütemezési feladatra, melynek speciális tulajdonsága az, hogy a rendelkezésre álló gépek (erőforrások) száma függ az időtől.

3.2. Az ütemezési probléma transzformációja

Az eredeti ütemezési probléma transzformációjának lényege a következő:

1. Az erőforrások rendelkezésre állási időintervallumait (a rögzített műszakokat) a műhelyhez rendelt globális rendszerben a kezdési időpont szerint növekvő sorrendben haladva besorszámozzuk decimális egészekkel (2. ábra). Ezeket a sorszámokat lépéseknek (step) nevezzük és s szimbólummal jelöljük. A legnagyobb kiosztott sorszám az s_{max}. A sorszámozás logikájából az következik, hogy a lépések összefüggő sorozatot alkotnak, így az új ütemezési modellben ez helyettesíti a valós időtengelyt.
2. A szerelő szakmunkásokból álló csoportokat párhuzamosan működő absztrakt (virtuális) gépekkel (erőforrásokkal) modellezzük. Egy virtuális gép egyszerre csak egy munkán dolgozhat, és minden egyes munkán egyszerre csak egy virtuális gép dolgozhat. A virtuális gépek száma az s lépések függvényében (műszakról műszakra) változhat. Ezt a változó kapacitáskorlátot szimbolikusan a $P(s)$ jelöléssel fejezzük ki. Az adott s lépésben rendelkezésre álló virtuális

gépek konkrét számát az érintett műszakban elvégezhető munkák számára vonatkozó eredeti korlátozás állítja be.

3. Az előírt konfiguráció-előkészítési tevékenységeket N_J számú egymástól független J_i $(i = 1, 2, \ldots, N_J)$ munkának tekintjük.
4. A J_i munkához tartozó valós időadatokat rendre átalakítjuk diszkrét s lépésekre (műszaksorszámokra).
   - 4.1. A konfigurációk valós előkészítési idejét a virtuális gépek műveletvégző képességéhez és darabszámához igazítjuk. Ennek következtében minden egyes munka műveleti ideje definíció szerint egy műszak hosszának megfelelő egységnyi értéket kap: $p_i := 1$ $(i = 1, 2, \ldots, N_J)$. Egy munka mindig egy kijelölt műszak elején kezdődik és a végén fejeződik be egy adott virtuális gépen.
   - 4.2. A J_i munka legkorábbi indítási időkorlátját egy megfelelően kiválasztott műszak diszkrét lépésszámával (r_i) definiáljuk. Az a legkorábbi műszak jelenti a korlátot, melynek már az elején minden indítási feltétel teljesül.
   - 4.3. A J_i munka legkésőbbi befejezési időkorlátját szintén egy megfelelően kiválasztott műszak diszkrét lépésszámával (d_i) definiáljuk. Itt az a legkésőbbi műszak jelenti a korlátot, melynek végéig be kell fejezni a munkát.
   - 4.4. A J_i munka teljesítésére kijelölt időintervallumot a hozzá tartozó diszkrét lépésszámmal fejezzük ki (C_i). Ez az ütemezési feladat döntési változója. A J_i munka végrehajtása a C_i döntési változó értéke által kijelölt műszak elején kezdődik és ugyanannak a műszaknak a végén fejeződik be valamelyik virtuális gépen.
   - 4.5. A J_i munka késését $L_i = C_i - d_i$; formalizmussal, valamint csúszását $T_i = max(0, L_i)$; formalizmussal szintén egységnyi lépésben mérjük. Két aktív műszak lépésben mért távolsága a hozzájuk tartozó sorszámok különbségeként adódik. Ez eltér a tényleges időpontok között eltelt időtől. Például, ha hétvégén nincs aktív műszak, akkor a péntek 14:00–22:00 óra közötti műszak és a hétfő 6:00–14:00 óra közötti műszak távolsága 1 lépés. Ha hétvégén van két aktív műszak, akkor az előbb említett pénteki és hétfői műszakok távolsága már 3 lépés.

A transzformációval előállított új ütemezési feladatot – a fenti jelölésekre alapozva és felhasználva a szakirodalomban használt α | β | γ formalizmust – a következőképpen definiáljuk:

$$P(s) \mid p_i = 1;\ r_i = integer;\ d_i = integer\ |L_{max} \tag{1}$$

Ez a feladattípus hasonló a Gharbi és Haouari által a [6] cikkben vizsgált feladathoz abban az értelemben, hogy mindkét modellben párhuzamos erőforrások szerepelnek és a munkáknak indítási és befejezési időkorlátai is vannak. Azonban a mi modellünkben minden egyes gépnek (erőforrásnak) saját rendelkezésre állási intervallumlistája van, míg a Gharbi-modell csupán egyetlen időablakot rendel minden géphez. A két modellben alkalmazott célfüggvények is eltérőek.

Brucker [4] könyvében korábban bemutatta a $P \mid p_i = 1 ; r_i = integer \mid L_{max}$ feladatot. A Brucker-féle modell nagyon hasonló az általunk definiált modellhez, mert abban is egységnyi műveleti idők, egész értékű indítási és befejezési időkorlátok szerepelnek. Azonban mi még azt is figyelembe vesszük, hogy az erőforrások (gépek) rendelkezésre állása időben változó lehet, így a kiterjesztett $P(s) \mid p_i = 1; r_i = integer; d_i = integer \mid L_{max}$ feladattípus speciális esetként magában foglalja a Brucker-féle klasszikus $P \mid p_i = 1; r_i = integer \mid L_{max}$ feladatot, mert abban a gépek száma kötött és minden gép folyamatosan rendelkezésre áll.

4. Az ütemezési feladat megoldása

4.1. Időtartalék-orientált ütemezési algoritmus

A $P(s) \mid p_i = 1; r_i = integer; d_i = integer \mid L_{max}$ ütemezési feladat megoldására kifejlesztettünk egy időtartalék-orientált felépítő jellegű ütemezési algoritmust. A módszer részleteit a 2. ábra szemlélteti.

Az időtartalék-orientált algoritmus leírásában használt eddig még nem definiált segédváltozók és azok jelentése a következő:

- *item*: ciklusváltozó, amely a munkák kiválasztásának iterációit különbözteti meg.
- *mach*: segédváltozó, amely egy adott műszakban elérhető virtuális gépek számlálására szolgál.
- R: halmaz, amely tartalmazza az adott műszakban ütemezhető munkákat.
- L_i: a beütemezett munka késése.
- T_i: a beütemezett munka csúszása.

A 2. ábrán látható megoldó algoritmus működési elve röviden a következő:

A diszkrét műszaksorszámokra transzformált időtengely mentén a változó számú műveletvégző kapacitás adott azáltal, hogy minden egyes s lépéshez (műszaksorszámhoz) tartozó $P(s)$ kapacitáskorlát értékét tároljuk egy vektorban. Induláskor rendezzük a munkákat a legkorábbi indítási időkorlát r_i értéke szerint nem csökkenő sorrendbe előkészítve a mindenkori aktuális s sorszámú műszakban indítható munkák halmazának lekérdezését. Majd a legkisebb r_i értéktől kezdve az s és $P(s)$ párok mentén haladva kiválasztjuk az aktuális beütemezendő munkát. A kiválasztás úgy történik, hogy az aktuális s értéket vezérlő paraméternek tekintve meghatározzuk az s műszakban indítható munkák halmazát. Ezt jelöli az R. Majd ebből az R halmazból kiválasztjuk a legkisebb időtartalékkal (slack) rendelkező munkát. Mivel minden egyes munka azonos műveleti idejű ebben a feladatban, így a legkisebb időtartaléka a legkorábbi határidejű munkának van. Az így kiválasztott munkát (J_i) a lehető legkorábbi szabad időintervallumra ütemezzük, tehát az aktuális s-re $(C_i \leftarrow s)$.

```
Időtartalék-orientált algoritmus a P(s) | pi = 1; ri = integer; di = integer | Lmax
feladathoz
{
  Rendezzük a Ji (i = 1, 2, …, NJ) munkákat ri szerint nem csökkenő sorrendbe;
  item ← 1;
  while (item ≤ NJ)
    {
      s ← ritem;
      while (P(s) < 1)
        {
          s ← s + 1;
          if (s > smax) Kilépés megvalósítható megoldás nélkül;
        }
      R ← {Ji | Ji nem ütemezett és ri ≤ s};
      mach ← 1;
      while (R nem üres)
        {
          Válasszuk ki a legkisebb di határidővel rendelkező Ji munkát az R-ből;
          R ← R \ {Ji};
          Ütemezzük a Ji-t a mach.-edik rendelkezésre álló gépre az s műszakban;
          Ci ← s;
          Li ← Ci - di;
          Ti ← max(0, Li);
          item ← item + 1;
          if (mach + 1 ≤ P(s)) mach ← mach + 1;
          else
            {
              mach ← 1;
              s ← s + 1;
              while (P(s) < 1)
                {
                  s ← s + 1;
                  if (s > smax) Kilépés megvalósítható megoldás nélkül;
                }
              R ← R ∪ {Ji | Ji nem ütemezett és ri ≤ s}
            }
        }
    }
  Visszatérés az elkészített optimális megoldással;
}
```

2. ábra. Időtartalék-orientált ütemezési algoritmus

Az algoritmus azokat a szituációkat is helyesen kezeli, amikor bizonyos műszakokban egyetlen munkát sem lehet elvégezni, mert ott a *P(s)* érték rendre nulla (nincs műveletvégző kapacitás). Az ilyen műszakokat az algoritmus nem terheli, hanem átlépi. Az is előfordulhat kapacitáshiány miatt, hogy nem minden aktivált munka ütemezhető be! Ezt jelzi a „Kilépés megvalósítható megoldás nélkül;" felirat.

Ha minden munka beütemezhető (létezik megvalósítható megoldás), akkor a bemutatott algoritmus minimális késést eredményező optimális megoldást állít elő polinomiális futási idő alatt. Ha a munkák sorszámai eleve az indíthatósági időpontok növekvő sorrendjében vannak kiosztva ($r_1 \leq r_2 \leq \ldots \leq r_i \leq \ldots \leq r_{N_J}$), akkor az algoritmus futási idejének ordója csupán $O(n \log n)$, ahol $n=N_J$ a munkák számát jelenti.

4.2. Az optimum elérésének bizonyítása

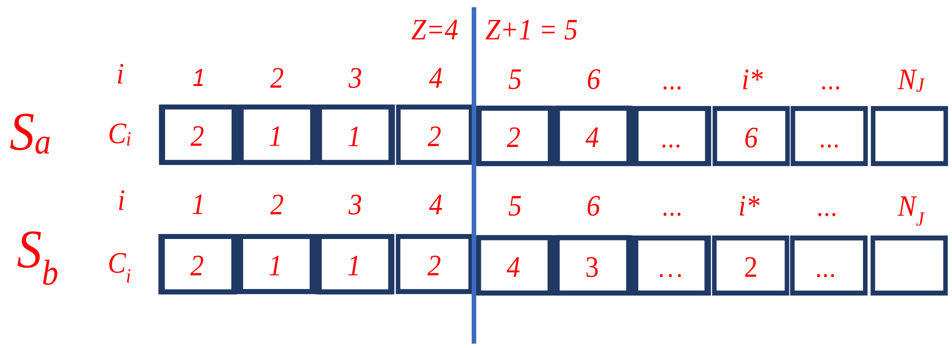
Az időtartalék-orientált algoritmus a $P(s) \mid p_i = 1; r_i = integer; d_i = integer \mid L_{max}$ ütemezési feladat optimális megoldását állítja elő.

A bizonyítás során bemutatjuk, hogy létezik olyan optimális megoldás, amelynek célfüggvény-értéke azonos az időtartalék-orientált algoritmus által előállított megoldás célfüggvény értékével.

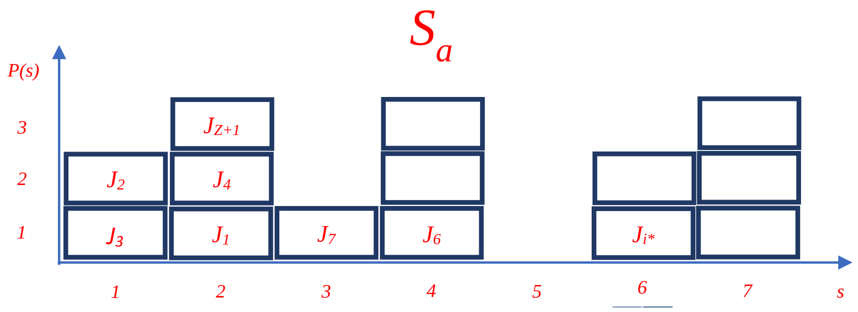
Legyenek a munkákhoz tartozó sorszámok (az *i* indexek) a megengedett legkorábbi indítási időpontok növekvő sorrendjében kiosztva: J_i $(i = 1, 2, \ldots, N_J)$, ahol $r_1 \leq r_2 \leq \ldots \leq r_i \leq \ldots \leq r_{N_J}$.

Jelölje S_a az időtartalék-orientált algoritmus által adott megoldást. Legyen S_b egy optimális megoldás.

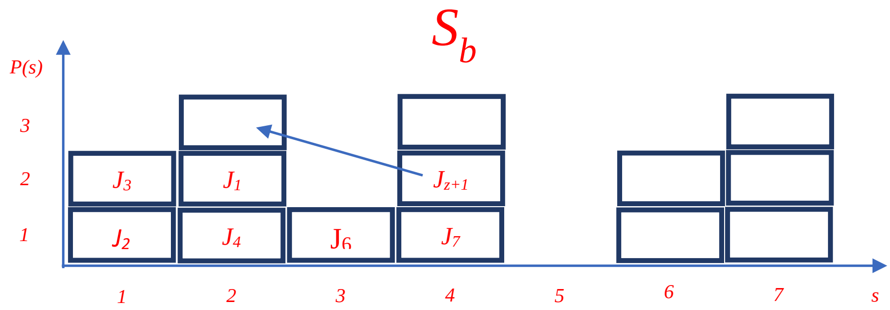
Az S_a és az S_b ismeretében meghatározható, hogy az *i* index szerinti első Z $(0 \leq Z \leq N_J)$ számú munka rendre ugyanabba a műszakba van beütemezve az S_a és az S_b szerint. Így $Z+1$ jelöli azt a legkisebb i indexet, amelynél a J_i munka C_i befejezési időpontja különbözik az S_a és az S_b szerint (3. ábra).



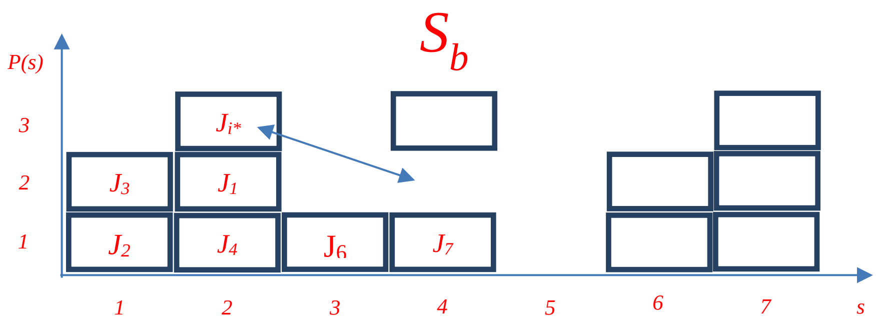
3. ábra. A *Z* érték jelentése két megoldás összehasonlításakor



4. ábra. Illusztratív példa az időtartalék-orientált algoritmus eredményére (S_a)



5. ábra. Illusztratív példa az első esetre



6. ábra. Illusztratív példa a második esetre

Ha $Z = N_J$, akkor minden egyes munka befejezési időpontja rendre azonos a két megoldás szerint, így a két megoldás célfüggvény-értéke azonos, továbbá mivel S_b optimális megoldás, így az S_a is optimális.

Az összehasonlítás módjából és az időtartalék-orientált algoritmus működéséből következik, hogy ha a $Z < N_J$, akkor a J_{Z+1} munka a C_{Z+1} műszakba van beütemezve az S_a szerint (4. ábra), míg ugyanez a J_{Z+1} munka a C_{Z+1} műszaknál későbbi műszakba van beütemezve az S_b szerint.

Ilyenkor két eset különböztethető meg attól függően, hogy van vagy nincs szabad virtuális gép a C_{Z+1} műszakban az S_b szerint:

1. Van szabad virtuális gép a C_{Z+1} műszakban az S_b szerint (5. ábra).
   Ebben az esetben a J_{Z+1} munka a szabad gépre áthelyezhető. Ez az S_b megoldáson elvégzett módosítás nem növeli meg az L_{max} célfüggvény értékét, mert a J_{Z+1} munka az eredetinél korábbi műszakba kerül, vagyis a késése nem nőhet. Tehát a módosított megoldás is optimális megoldás.

2. Nincs szabad virtuális gép a C_{Z+1} műszakban az S_b szerint (6. ábra).
   Ebben az esetben létezik egy J_{i^*} munka, amelyik a C_{Z+1} műszakba van beütemezve az S_b szerint, de ez a J_{i^*} munka a C_{Z+1} műszaknál későbbi műszakba van beütemezve az S_a szerint.
   Mivel az időtartalék-orientált algoritmus a J_{Z+1} munkát korábbi kezdéssel ütemezte be az S_a szerint mint a J_{i^*} munkát, így az algoritmus működéséből adódik, hogy a J_{i^*} munka d_{i^*} határideje nagyobb vagy egyenlő mint a J_{Z+1} munka d_{Z+1} határideje: $d_{i^*} \geq d_{Z+1}$. Bevezetve az $s_t \geq 0$ értéket, írható hogy:

$$d_{i^*} = d_{Z+1} + s_t \tag{2}$$

Felhasználva a késés $L_i = C_i - d_i$ definícióját, az érintett munkák domináns késése az S_b megoldás szerint a következőképpen alakul:

$$\begin{aligned} max(L_{i^*}, L_{Z+1}) &= max(C_{i^*} - d_{i^*}, C_{Z+1} - d_{Z+1}) = \\ &= max(C_{i^*} - d_{Z+1} - s_t, C_{Z+1} - d_{Z+1}). \end{aligned} \tag{3}$$

Mivel $C_{Z+1} > C_{i^*}$, bevezetve az $s_T > 0$ értéket, írható hogy:

$$C_{Z+1} = C_{i^*} + s_T. \tag{4}$$

Ezt felhasználva adódik, hogy:

$$max(L_{i^*}, L_{Z+1}) = max(C_{i^*} - d_{Z+1} - s_t, C_{i^*} + s_T - d_{Z+1}). \tag{5}$$

Bevezetve az $L_{iZ} = C_{i^*} - d_{Z+1}$ jelölést adódik, hogy:

$$max(L_{i^*}, L_{Z+1}) = max(L_{iZ} - s_t, L_{iZ} + s_T) = L_{iZ} + s_T. \tag{6}$$

Az S_b megoldás J_{Z+1} és a J_{i^*} munkájának felcserélése után a két munka domináns késése a következőképpen alakul:

$$max(L_{i^*}, L_{Z+1}) = max(C_{Z+1} - d_{i^*}, C_{i^*} - d_{Z+1}). \tag{7}$$

Behelyettesítve a bevezetett jelöléseket adódik, hogy:

$$max(L_{i*}, L_{Z+1}) = max(C_{i*} + s_T - d_{Z+1} - s_t, C_{i*} - d_{Z+1}) =$$
$$= max(L_{iZ} + s_T - s_t, L_{iZ}) \quad (8)$$

$$max(L_{i*}, L_{Z+1}) = L_{iZ} + s_T - s_t. \quad (9)$$

Mivel S_b optimális, így a csere után sem lehet kisebb a célfüggvény értéke mint az S_b esetében, ebből következik hogy:

$$L_{iZ} + s_T - s_t \geq L_{iZ} + s_T. \quad (10)$$

Mivel $s_t \geq 0$, így a J_{Z+1} és a J_{i*} munka felcserélése az S_b megoldásban nem növeli meg a célfüggvény értékét.

Mindkét lehetséges esetben az eredeti optimális S_b megoldásból kiindulva a módosított S_b megoldás is optimális maradt, ugyanakkor most már az első $Z+1$ darab munka fejeződik be azonos műszakban az S_a és a módosított S_b szerint. A módosítás hatására a kiinduláshoz képest a Z érték növekedett eggyel.

A fenti logikát folytatva a módosítások sorozata elvezet oda, hogy a Z értéke addig nő, míg végül minden egyes J_i munka (i = 1, 2, …, N_J) rendre ugyanabba a C_i műszakba lesz beütemezve a vizsgált S_a és egy optimális (többszörösen módosított S_b) megoldás szerint. Ebből az következik, hogy az S_a megoldás is optimális. Ez egyértelműen bizonyítja, hogy az időtartalék-orientált algoritmus a célfüggvény szerint optimális megoldást állít elő.

5. Összefoglalás és következtetések

A cikkben bemutatott ütemezési feladat megoldására kidolgoztunk egy új modellt, mely egy időben változó kapacitáskorlátos erőforrás-környezetet reprezentál, ahol egymástól független egyetlen operációt igénylő munkákat kell végrehajtani. A munkákhoz szigorú indítási időkorlátok, befejezési határidők és egységnyi műveleti idők kapcsolódnak. Az ütemezés elsődleges célja a határidők betartása.

A legnagyobb késés minimalizálására kidolgoztunk egy új időtartalék-orientált ütemezési algoritmust, amely polinomiális futási idő alatt optimális megoldást állít elő. Az optimális megoldás ismeretében objektíven megválaszolható az a kérdés, hogy a vizsgált ütemezési feladatban szereplő munkák mindegyike elkészíthető-e a saját határidejére.

A konkrét probléma sikeres megoldásán túlmenően általánosítható eredmények születtek. A kidolgozott konkrét modell és a kapcsolódó megoldási módszer hatékonyan alkalmazható különböző ütemezési feladatok megoldására.

Az egyik lehetséges eset az, amikor az ütemezendő folyamat közvetlenül kezelhető a modellel. A gyártási főfolyamatok és segédfolyamatok sokféleségéből következik, hogy számos tevékenység egységnyi műveleti idővel modellezhető (pl.: rutinszerű mérési/ellenőrzési folyamat, csomagolás, szerszám-előkészítés, belső logisztikai folyamatelemek stb.), így ezekben az esetekben a modell közvetlenül alkalmazható, akár van változó kapacitáskorlát, akár rögzített kapacitáskorlátot kell figyelembe venni. A felsorolt példák alapvetően diszkrét gyártási folyamatokra vonatkoztak, ugyanakkor a problémát általánosabban szemlélve számos szolgáltatással kapcsolatos ütemezési feladat is idesorolható. Ilyen feladat lehet például a nagy értékű, korlátozottan rendelkezésre álló orvosi műszerek és diagnosztizáló személyzet időkapacitásának figyelembevétele határidős, sorban álló beavatkozások és/vagy vizsgálatok ütemezése során. További alkalmazási terület lehet például a nagy méretű logisztikai rendszerek kiszolgálásakor jelentkező, dinamikusan generált beépülő feladatok (pl.: repülőtéri üzemanyag-feltöltési tevékenységek, kikötői konténerrakodási műveletek stb.) ütemezése.

Az eredmények hasznosításának egy másik lehetséges módja az, amikor a konkrét modell és annak algoritmusai nem közvetlenül kerülnek bevetésre, hanem közvetett módon kerülnek alkalmazásba. Például, ha valamely folyamatban a műveleti időket nem célszerű vagy nem is lehet egységnyi értéknek tekinteni, a bemutatott koncepció akkor is alkalmazható, de már nem garantálható az optimális megoldás elérése. Ilyenkor a bemutatott módszerek gyors heurisztikus felépítő algoritmusoknak tekinthetők. Az NP-hard feladatok esetében ezeknek nagyon fontos szerepe van. Ezt figyelembe véve, a korlátozottan rendelkezésre álló párhuzamos kapacitásokból és az egyoperációs határidős munkákból álló ütemezési feladatok halmaza tovább szélesedik.

Az időtartalék-orientált szemlélet használható több operációból álló munkák esetében is. Ilyenkor az adott munka indítási időpont-korlátja az elsőként indítható operációra vonatkozik, míg a határidő az utolsó operáció befejezési idejének szab felső határt. Ez alapján minden egyes operáció esetében számítható egy saját időtartalék úgy, hogy a kapcsolódó munka határidejéből ki kell vonni ennek a munkának a még el nem végzett operációinak műveleti időösszegét, továbbá ki kell vonni a döntés pillanatához tartozó mindenkori aktuális időpontot is. Ezt felhasználva az időtartalék-orientált algoritmus úgy működik, hogy a változó kapacitáskorlát mentén haladva ki kell választani a vizsgált időpontban indítható operációkat a teljes operációhalmazból, majd a legkisebb időtartalékkal rendelkező operációt kell beütemezni a lehető legkorábbi időpontra. Ha a kapacitáskorlátot eléri a terhelés, akkor tovább kell lépni az időben a következő szabad kapacitás rendelkezésre állásáig.

Az így általánosított elvek önállóan vagy más heurisztikus módszerekkel kombinálva hatékonyan alkalmazhatók erőforrás-korlátos és határidős összetett ütemezési feladatok megoldására. Például az ismert klasszikus erőforrás-környezetek (flow shop, job shop, open shop, stb.) a bemutatott időben változó kapacitáskorlátokkal kiegészítve további új kiterjesztett modelleket alkotnak.

Irodalom

[1] Allahverdi, A., Ng, C. T., Cheng, T. C. E., Kovalyov, M. Y.: A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, Vol. 187, 2008, pp. 985–1032., https://doi.org/10.1016/j.ejor.2006.06.060.

[2] Aytug, H., Lawley, M. A., Mckay, K., Mohan, S., Uzsoy, R.: Executing production schedules in the face of uncertainties: a review and some future directions. *European Journal of Operational Research*, Vol. 161, 2005, pp 86–110. https://doi.org/10.1016/j.ejor.2003.08.027

[3] Baykasoğlu, A., Özbakir, L., Dereli, T.: Multiple dispatching rule based heuristic for multi-objective scheduling of job shops using tabu search. *Proceedings of the 5th International Conference on Managing Innovations in Manufacturing*, Milwaukee, USA, 2002, pp. 396–402.

[4] Brucker, P.: *Scheduling Algorithms*. 5th ed., Springer-Verlag Berlin Heidelberg, 2007.

[5] Cheng, T. C. E., Sin, C. C. S.: A state-of-the-art review of parallel-machine scheduling research. *European Journal of Operational Research*, Vol. 47, 1990, pp. 271–292. https://doi.org/10.1016/0377-2217(90)90215-W

[6] Gharbi, A., Haouari, M.: Optimal parallel machines scheduling with availability constraints. *Discrete Applied Mathematics*, Vol. 148, 2005, pp. 63–87. https://doi.org/10.1016/j.dam.2004.12.003

[7] Koulamas, C.: The total tardiness problem: review and extensions, Operations Research, Vol. 42, 1994, pp. 1025–1041., https://doi.org/10.1287/opre.42.6.1025.

[8] Kulcsár, Gy., Erdélyi F.: A New Approach to Solve Multi-Objective Scheduling and Rescheduling Tasks. *International Journal of Computational Intelligence Research*, 3 (4), 2007. pp. 343–351.

[9] Lei, D.: Multi-objective production scheduling: a survey. *The International Journal of Advanced Manufacturing Technology*, Vol. 43, Issue 9–10, 2009, pp. 926–938. https://doi.org/10.1007/s00170-008-1770-4

[10] Loukil, T., Teghem, J., Tuyttens, D.: Solving multi-objective production scheduling problems using metaheuristics. *European Journal of Operational Research*, 161, 2005, pp. 42–61., https://doi.org/10.1016/j.ejor.2003.08.029.

[11] Mokotoff, E.: Parallel machine scheduling problems: a survey. *Asia-Pacific Journal of Operational Research*, Vol. 18, 2001, pp. 193–242.

[12] Pinedo, M. L.: *Planning and Scheduling in Manufacturing and Service*. 2nd ed., Springer Verlag, New York, 2009, https://doi.org/10.1007/978-1-4419-0910-7_12.

[13] Pinedo, M. L.: *Scheduling Theory, Algorithms, and Systems*. 5th ed., Springer Verlag, New York, 2015, https://doi.org/10.1007/978-3-319-26580-3.

[14] Sbalzarini, L. F., Müller, S., Koumoutskos, P.: Multiobjective optimization using evolutionary algorithms. In *Center of Turbulence Research, Proceedings of the Summer Program 2000*, pp. 63–74.