ŐSZ OLIVÉR[1]

Ciklikus üzemű robotkarok ütemezése

1. BEVEZETÉS

A gyártási folyamatok optimalizációja egyre fontosabb szerepet kap napjainkban, és az Ipar 4.0 terjedésével új kihívások és lehetőségek jelennek meg ezen a kutatási területen. Egyre szélesebb körben terjed el az automatizálás, ami olyan mértékben növeli a gyártás hatékonyságát, amely emberi erőforrással elérhetetlen. A gyártáshoz használt robotok kiszámíthatóbbak és bonyolultabb ütemezéseket is pontosan tudnak követni. Hatékony felhasználásukhoz azonban pontosan meg kell tervezni a gyártási folyamatok időzítését.

A robotok használata olyan rendszerekben a leggazdaságosabb, ahol a piaci igények stabilak, így hosszabb ideig nem kell változtatni a gyártási folyamatokon. Ilyen rendszerekben hosszútávú ütemezési feladatok merülnek fel, azaz több hónapra előre meg kell határozni a gyártási tervet. Ilyen nagyméretű feladatokat nem lehet közvetlenül megoldani egzakt (optimalitást garantáló) módszerekkel. Vagy valamilyen heurisztikus módszert kell használni, vagy le kell egyszerűsíteni a feladatot. Egy gyakran használt egyszerűsítés a ciklikus ütemezés, mely során egy rövidebb időtartam (1–5 nap) kerül ütemezésre, melynek megoldását – a ciklust – periodikusan ismétlődve fogják végrehajtani a hosszabb időtartam alatt.

Jelen dolgozatban egy új kevert-egész lineáris programozási (MILP) modell kerül bemutatásra, mellyel általános ciklikus ütemezési feladatokat lehet megoldani, azaz nem csak robotkarok ütemezésére használható. A 2. fejezet részletesen ismerteti a feladatosztály jellemzőit és szakirodalmát. A 3. fejezetben található a kifejlesztett matematikai modell, majd a 4. fejezet bemutatja az új módszerrel végzett tesztek eredményét. Végül az 5. fejezet összefoglalja a munka eredményeit és a jövőbeli kutatási terveket a témában.

2. A CIKLIKUS ÜTEMEZÉSI FELADAT

A hosszútávú ütemezés során általában egy megadott időtartam alatt elért profitot kell maximalizálni, vagy adott nagy mennyiségű terméket kell előállítani minél rövidebb idő alatt. A ciklikus megoldásban mindkét esetből az következik, hogy az egy ciklus alatt legyártott termékmennyiség (vagy az ebből származó

[1] SZE Multidiszciplináris Műszaki Tudományi Doktori Iskola, PhD hallgató. Témavezető: Dr. Hegyháti Máté, Gépészmérnöki, Informatikai és Villamosmérnöki Kar, Informatika Tanszék, tudományos főmunkatárs.

profit) és a ciklusidő hányadosát kell maximalizálni. A ciklusidő a két egymást követő ciklus kezdete között eltelt idő.

A célfüggvény tehát egy olyan hányados, melyben a számláló és a nevező értéke is a döntési változóktól függ. Ez még rövidtávú ütemezés esetén is nehezen, csak magas számítási igényű nemlineáris optimalizálási módszerekkel kezelhető. Ezek a nagy számú diszkrét változó miatt nem képesek hatékonyan megoldani az ütemezési feladatokat. Emiatt a szakirodalomban gyakran a számláló vagy a nevező különböző diszkrét értékekre való rögzítése mellett maximalizálják a hányadost. Gyakorlati szempontok miatt általában a termékmennyiséget rögzítik, így az egyes termékek aránya a piaci igényekhez igazítható. Az itt bemutatott módszer is ezt a technikát alkalmazza. A másik megoldást, a ciklusidő rögzítését használó módszerre egy példa Hegyháti és Ősz[2] modellje.

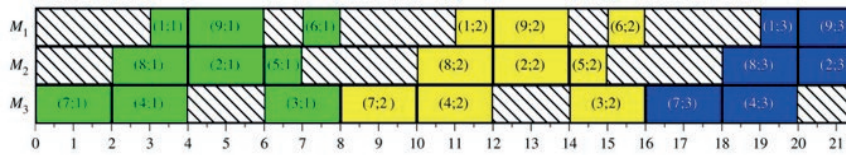
Ütemezési feladatoknak sokféle típusa ismert, melyek eltérő döntési lehetőségekkel és korlátozásokkal rendelkeznek. Jelen tanulmány a szakirodalomban job-shop ütemezési feladatként ismert feladatosztály ciklikus változatával foglalkozik. Ebben a feladatban a termékek előállításához szükséges gyártási lépéseket (műveleteket) megadott sorrendben kell végrehajtani, és minden műveletet egy adott berendezés képes elvégezni. Ugyanaz a berendezés több termék műveletéhez is kellhet, de egyszerre csak egy műveletet tud végrehajtani, megadott idő alatt. Emiatt a megoldás során meg kell határozni, hogy egyes időpontokban mely műveleteket kell végeznie.

A klasszikus job-shop feladatban a cél adott mennyiségű termékre a befejezési idő minimalizálása. Az így meghatározott ütemezés is egy lehetséges megoldása a ciklikus ütemezési feladatnak, ahol a befejezési idő lesz a ciklusidő, ami után megismételhető az ütemezés. Azonban a ciklikus működés miatt jobb megoldás érhető el, ha azon berendezések, melyek már befejezték műveleteiket, elkezdik a következő ciklus műveleteit, amíg más berendezések még dolgoznak. Ez a Brucker és Kampmeyer[3] által machine-chain repetitionnek nevezett megoldás, melyben az egyes berendezéseknek saját ciklusuk van, így ezek kezdetét egymáshoz képest el lehet tolni, de hosszuk megegyezik, ez lesz a ciklusidő. A különbség megfigyelhető az alábbi ábrákon, melyek B&K cikkéből származnak. Az 1. ábrán a klasszikus job-shop ütemezése ismétlődik, a 2. ábrán a machine-chaint használó megoldás látható.

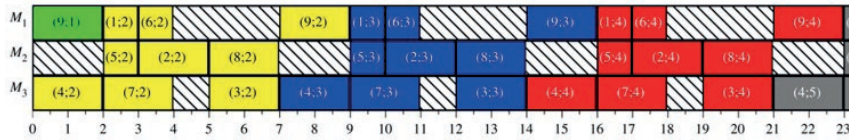
[2] Hegyháti, Máté – Ősz, Olivér: Interlacing in cyclic scheduling, in *VOCAL 2018 8th VOCAL Optimization Conference: Advanced Algorithms: Short Papers,* 2018, pp. 50–55.

[3] Brucker, Peter – Kampmeyer, Thomas: A general model for cyclic machine scheduling problems, in *Discrete Applied Mathematics*, 2008, 156/11., pp. 2561–2572.

1. ábra: Ciklusok a job-shop ütemezés ismétlésével

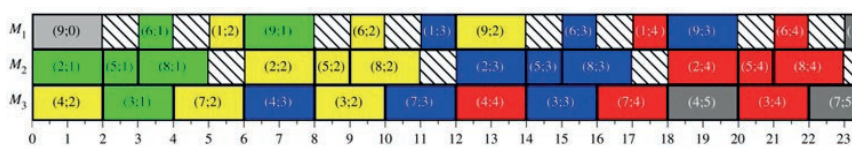


2. ábra: Ciklusok machine-chain repetitionnel



Az erőforráskihasználtságot még tovább lehet növelni, ha megengedjük az egymást követő ciklusok átlapolódását, azaz interlacinget. Ez azt jelenti, hogy egy berendezés elkezdheti a következő ciklusához tartozó műveleteket, mielőtt az összes műveletét befejezte volna, pl. mert más berendezések kimenetére várakozik. Ebben az esetben fontos megkülönböztetni a ciklusidőt a ciklus hosszától: előbbi a ciklusok kezdete között eltelt idő, utóbbi pedig az egy ciklusra meghatározott termékmennyiség gyártási ideje az első művelettől az utolsóig. A 3. ábra az előző példának az átlapolódást használó megoldását mutatja be.

3. ábra: Átlapolódó ciklusok



Megfigyelhető, hogy az átlapolódás megengedésével a ciklusidő 7-ről 6 időegységre csökkent, viszont a ciklus hossza 9-ről 14-re nőtt. Tehát több termék készül el adott idő alatt, de egy termék több időt tölt a rendszerben, mire elkészül. Emiatt általánosságban több tárolóra van szükség, hogy a műveletek között tárolni tudjuk az anyagokat. Ha nem korlátozzuk az átlapolódást, akkor a feladat leegyszerűsödik egy open-shop feladatra, ami a dedikált berendezések miatt hozzárendelési döntést nem igényel, így polinomiális időben megoldható, ahogy azt Bożejko és társai[4] bizonyították. Azonban az így kapott megoldás a legtöbb valós rendszerben nem megvalósítható a felmerülő hatalmas tárkapacitási igény miatt.

[4] Bożejko, Wojciech – Gnatowski, Andrzej – Pempera, Jarosław – Wodecki, Mieczysław: Parallel tabu search for the cyclic job shop scheduling problem, in *Computers & Industrial Engineering*, 2017, Vol. 113., pp. 512–524.

Kétféle korlátozással is lehet mérsékelni az átlapolódás által megnövekedő tárigényt. Az egyik a blocking, ami úgy modellezi a rendszert, hogy nem áll rendelkezésre köztes tároló, így az egyes műveletek kimenetei addig a berendezésben maradnak – és blokkolják azt más művelet végzésétől –, amíg át nem kerülnek a következő műveletet végrehajtó berendezésbe. A másik korlátozási lehetőség, hogy limitáljuk azt, hogy hány egymást követő ciklus lapolódhat át egymással. Ezt a felső korlátot a ciklus magasságának nevezik, jele: H.

B&K modellje mindkét korlátozást tudja kezelni, de a korlátozásokban megjelenik két változó szorzata, így lineáris optimalizálási módszerekkel nem oldható meg. A szerzők egy másik cikkükben[5] részletesen bemutatják, hogy a modellt egy tabu keresés alapú heurisztikus módszerben alkalmazzák az ütemezési feladat megoldására. A következő fejezetben bemutatásra kerülő modell B&K modelljének továbbfejlesztése, és nagy előnye, hogy lineáris, így bármilyen MILP megoldó program segítségével meghatározható vele az optimális ütemezés.

3. MATEMATIKAI MODELL

A cél a ciklusidő ($\alpha \geq 0$) minimalizálása, ahogy (1) mutatja.

$$\min \alpha \qquad (1)$$

A ciklusok átlapolódását a H paraméter korlátozza. Az átlapolódások kezeléséhez a modell H db ciklus műveleteit ütemezi. H=1 esetén nincs átlapolódás, H=2 esetén csak a közvetlen egymást követő ciklusok lapolódhatnak át.

A műveletek kezdési és befejezési idejét a t^s és t^f változók jelölik, mindkettő nemnegatív. A termékek (azaz jobok) halmaza a J, minden termékhez adott egy b_j batch-szám (ennyi adag gyártandó egy ciklusban), és egy s_j lépésszám (ennyi műveletből áll). Így minden műveletet 4 index azonosít:

$$Op = \{(j,b,s,c) \mid j \in J,\ b \in 1 \dots b_j,\ s \in 1 \dots s_j,\ c \in 1 \dots H\}$$

A műveletek végrehajtási idejét $p_{j,s}$ jelöli. A blocking műveletek viszont addig blokkolják a végrehajtó berendezést ($m_{j,s}$), amíg meg nem kezdődik a termék következő lépése. Ezért a (2–5) korlátozások külön kezelik a blocking és non-blocking műveletek végrehajtását.

$$t^f_{j,b,s,c} \geq t^s_{j,b,s,c} + p_{j,s} \quad \forall (j,b,s,c) \in Op,\ (j,s) \text{ blocking} \qquad (2)$$

$$t^f_{j,b,s,c} = t^s_{j,b,s,c} + p_{j,s} \quad \forall (j,b,s,c) \in Op,\ (j,s) \text{ non-blocking} \qquad (3)$$

$$t^s_{j,b,s+1,c} = t^f_{j,b,s,c} + p_{j,s} \quad \forall (j,b,s,c) \in Op,\ s < s_{j,}\ (j,s) \text{ blocking} \qquad (4)$$

$$t^s_{j,b,s+1,c} \geq t^f_{j,b,s,c} + p_{j,s} \quad \forall (j,b,s,c) \in Op,\ s < s_{j,}\ (j,s) \text{ non-blocking} \qquad (5)$$

[5] Brucker, Peter – Kampmeyer, Thomas: Cyclic job shop scheduling problems with blocking, in *Annals of Operations Research*, 2008, 159/1., pp. 161–181.

Az egymást követő ciklusok műveletei között az időkülönbség **α**. Ezt határozzák meg a (6–7) korlátozások.

$$t^s_{j,b,s,c+1} - t^s_{j,b,s,c} = \alpha \quad \forall (j,b,s,c) \in Op,\ c < H \tag{6}$$

$$t^f_{j,b,s,c+1} - t^f_{j,b,s,c} = \alpha \quad \forall (j,b,s,c) \in Op,\ c < H \tag{7}$$

A ciklus hosszát az *L* segédváltozó jelöli. Alsó korlátját (8) adja meg, a (9) miatt pedig alulról korlátozza a célfüggvényt. A (9) korlátozás limitálja az átlapolódást: könnyen belátható, hogy ha $\alpha < \frac{L}{H}$, akkor legalább H új ciklus megkezdődik, mielőtt a jelenlegi befejeződne, hiszen **α** időközönként ismétlődnek.

$$L \geq t^f_{j,b,s,1} - t^s_{j',b',s',1} \quad \forall (j,b,s,1),(j',b',s',1) \in Op \tag{8}$$

$$\alpha \geq \frac{L}{H} \tag{9}$$

Az egy berendezéshez tartozó műveletek sorrendjét az *x* bináris precedencia változók állítják be. Ha $x_{a,b} = 1$, akkor *a* művelet végrehajtása megelőzi *b*-t. Az azonos berendezésekhez tartozó műveletek között muszáj meghatározni a sorrendet, mert egyszerre nem hajthatók végre. Az ilyen műveletpárokra (10) garantálja, hogy a precedencia reláció valamelyik irányban teljesüljön.

$$\begin{gathered} x_{j,b,s,c,j',b',s',c'} + x_{j',b',s',c',j,b,s,c} = 1 \\ \forall (j,b,s,c) \neq (j',b',s',c') \in Op,\ m_{j,s} = m_{j',s'} \end{gathered} \tag{10}$$

A precedencia relációban álló műveletpárok időzítését (11) korlátozza. Itt *M* egy kellően nagy értékű paraméter, pl. $M = \sum_{(j,b,s,c) \in Op} p_{j,s}$.

$$\begin{gathered} t^s_{j',b',s',c'} \geq t^f_{j,b,s,c} - M\left(1 - x_{j,b,s,c,j',b',s',c'}\right) = 1 \\ \forall (j,b,s,c) \neq (j',b',s',c') \in Op,\ m_{j,s} = m_{j',s'} \end{gathered} \tag{11}$$

A fenti korlátozások (1–11) elegendők az optimális megoldáshoz, azonban a (12–14) korlátozások javítják a hatékonyságot bizonyos esetekben.
A (12) beállítja, hogy az azonos ciklushoz tartozó műveletek sorrendje minden ciklusban megegyezzen. Ez a (6–7) korlátozásokból is következik, de segítheti a megoldást, ha a megfelelő bináris változókat is egyenlővé tesszük.

$$\begin{gathered} x_{j,b,s,c,j',b',s',c} = x_{j,b,s,c',j',b',s',c'} \\ \forall (j,b,s,c) \neq (j',b',s',c) \in Op,\ m_{j,s} = m_{j',s'},\ c' \in 1 \ldots H \end{gathered} \tag{12}$$

A másik gyorsítás a Holczinger és társai[6] által S-gráf alapú módszernél alkalmazott technika megvalósítása a modellben. Ha egy termékből több adagot kell gyártani, azok sorrendje tetszőleges. Ezáltal redundáns megoldások keletkeznek, melyek csak az azonos termékhez tartozó adagok sorrendjében térnek el. A (13-14) korlátozások rögzítik a köztük lévő sorrendet, ezáltal kiszűrik az efféle redundanciát.

$$\begin{gathered} t^{f}_{j,b,s,c} \leq t^{s}_{j,b',s',c'} \\ \forall j \in J,\ s,s' \in 1 \dots s_j,\ b,b' \in 1 \dots b_j,\ c,c' \in 1 \dots H, \\ s \leq s',\ b < b',\ c \leq c' \end{gathered} \tag{13}$$

$$\begin{gathered} x_{j,b,s,c,j,b',s',c'} = 1 \\ \forall (j,b,s,c) \neq (j,b',s',c') \in Op, \\ s \leq s',\ b < b',\ c \leq c' \end{gathered} \tag{14}$$

4. EMPIRIKUS TESZTEK

A bemutatott modell szakirodalomban elterjedt benchmark példákon lett tesztelve, melyek klasszikus job-shop feladatok átírása ciklikus ütemezésre. Az első példa (FT06) Fisher és Thompson[7] cikkéből származik, 6 terméket tartalmaz, melyek mind 6 lépésből állnak, melyek a 6 berendezést különböző sorrendekben érintik. A további 5 példa (LA01–LA05) Lawrence-től[8] származik, ezekben 10 termék van egyenként 5 művelettel, valamint 5 berendezés. Minden példa 4 változatban került tesztelésre: a műveletek blocking vagy non-blocking típusúak, miközben a H értéke 1 vagy 2.

A tesztekhez használt számítógép fő paraméterei: 2-magos Intel i5-660 CPU 3.33GHz, 8Gb RAM. A MILP megoldó program a Gurobi 8.0 verziója. A megoldási időkorlát 3600 s volt, ezután az addig megtalált legjobb megoldást adta meg a program.

A tesztek eredményeit az 1. táblázat foglalja össze. Az α oszlop a legjobb megtalált megoldás ciklusidejét mutatja, a CPU s pedig a futási időt. A non-blocking példákra, ahol időkorláton belül megtalálta a megoldó az optimális ütemezést, ott a ciklusidő megegyezik a szakirodalomból ismert optimum értékével, ami

[6] Holczinger, Tibor – Romero, Jessenia – Puigjaner, Luis – Friedler Ferenc: Scheduling of multipurpose batch processes with multiple batches of the products, in *Hungarian Journal of Industrial Chemistry*, 2002, 30/4., pp. 305–312.

[7] Fisher, Henry – Thompson, Gerald Luther: Probabilistic learning combinations of local job-shop scheduling rules, in Muth, John F. – Thompson, Gerald Luther – Winters, Peter R. (eds.): *Industrial scheduling*, 1963, Prentice Hall, Englewood Cliffs, pp. 225–251.

[8] Lawrence, S.: *Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques*, 1984, GSIA, Pittsburgh, Carnegie Mellon University.

igazolja a modell helyességét. A blocking műveleteket használó változatra viszont nem találtam a szakirodalomban közölt értékeket.

A futási időkből jól látható, hogy az átlapolódást megengedő (H=2) feladatot tovább tart megoldani. Ez nem meglepő, hiszen itt több változó van, nagyobb az ütemezés szabadsági foka. A ciklusidőn viszont sok esetben így is sikerült javítani az átlapolódással, a vastagon jelölt értékek jelölik a legjobb ciklusidőt külön a blocking és non-blocking változatra. Mivel a feladat hosszútávú ütemezésre vonatkozik, nem kell gyakran elvégezni az ütemezést, így a ciklusidő csökkentéséért megéri a hosszabb megoldási idő. Nagyobb méretű feladatokra viszont továbbra is célszerű heurisztikus módszereket alkalmazni. A MILP megoldók is alkalmaznak heurisztikákat, de ezek nem feladatspecifikusak, nem tudják kihasználni az ütemezési feladatok sajátosságait.

1. táblázat: Futási eredmények

Példa	H=1				H=2			
	non-blocking		blocking		non-blocking		blocking	
	α	CPU s	**α**	CPU s	**α**	CPU s	**α**	CPU s
FT06	55	0.5	63	0.64	43	1140.02	51	97.78
LA01	666	2079.21	793	131.66	666	3600	824	3600
LA02	655	3600	793	237.35	635	3600	802	3600
LA03	597	179.17	715	34.24	588	3600	687	3600
LA04	590	724.45	743	318.28	537	3600	666	3600
LA05	593	3600	664	321.47	593	3600	678	3600

5. ÖSSZEFOGLALÁS

Egy új MILP modell készült gyártórendszerek ciklikus ütemezéséhez, mely képes kezelni a ciklusok átlapolódását, és ezáltal jobb megoldást elérni. Erre a feladatosztályra ez az első egzakt lineáris modell. Nagyméretű példákra a nagy számítási igény miatt sokáig tart az optimális megoldás megtalálása, de ekkor is általában rövid idő alatt talál elfogadható megoldást a MILP megoldó.

A szakirodalomban tabu keresés heurisztikát alkalmaznak nagyobb méretű feladatok megoldására. Ehhez a módszerhez is szükség van egy jó modellre. A jövőben tervezem megvizsgálni, hogy a kifejlesztett MILP modell hatékonyan felhasználható-e tabu keresésben vagy más heurisztikus módszerben.

A bemutatott modell a klasszikus job-shop ütemezési feladat ciklikus változatát tudja kezelni. A feladatnak azonban számos változata van, különböző gyakorlatból származó korlátozásokkal. A szállító robotok kezelése is egy külön

feladatosztály, melyben a szállítások ideje függ a megtett távolságtól. Ennek egyszerűsített változatát már a jelenlegi modell is képes kezelni, de a jövőben tervezem kiterjeszteni további robotkar ütemezési feladatokra.

FELHASZNÁLT IRODALOM

- Bożejko, Wojciech – Gnatowski, Andrzej – Pempera, Jarosław – Wodecki, Mieczysław: Parallel tabu search for the cyclic job shop scheduling problem, in *Computers & Industrial Engineering*, 2017, Vol. 113., pp. 512–524.
- Brucker, Peter – Kampmeyer, Thomas: A general model for cyclic machine scheduling problems, in *Discrete Applied Mathematics*, 2008, 156/11., pp. 2561–2572.
- Brucker, Peter – Kampmeyer, Thomas: Cyclic job shop scheduling problems with blocking, in *Annals of Operations Research*, 2008, 159/1., pp. 161–181.
- Fisher, Henry – Thompson, Gerald Luther: Probabilistic learning combinations of local job-shop scheduling rules, in Muth, John F. – Thompson, Gerald Luther – Winters, Peter R. (eds.): *Industrial scheduling*, 1963, Prentice Hall, Englewood Cliffs, pp. 225–251.
- Lawrence, S.: *Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques*, 1984, GSIA, Pittsburgh, Carnegie Mellon University.
- Hegyháti, Máté – Ősz, Olivér: Interlacing in cyclic scheduling, in *VOCAL 2018 8th VOCAL Optimization Conference: Advanced Algorithms: Short Papers*, 2018, pp. 50–55.
- Holczinger, Tibor – Romero, Jessenia – Puigjaner, Luis – Friedler Ferenc: Scheduling of multipurpose batch processes with multiple batches of the products, in *Hungarian Journal of Industrial Chemistry*, 2002, 30/4., pp. 305–312.