Egylépéses gyártási feladatok költség optimális ütemezése időzített automatával

Vida Judit
Témavezető: Dr. Hegyháti Máté
Széchenyi István Egyetem

2018. február 27.

Tartalomjegyzék

1. fejezet

Bevezetés

Ütemezési problémákkal az élet számos területén találkozhatunk, hiszen hétköznapi feladatainkat is be kell osztanunk. Vannak azonban speciális problémák, amelyre már léteznek kidolgozott megoldások, mint például az S-gráf és a MILP modellek, ezen kívül adottak egyéb megoldási lehetőségek is, mint az időzített automaták. Dolgozatomban egy gyártási probléma ütemezését választottam, amelyet költségfüggvénnyel kiterjesztett időzített automata használatával optimalizálok, és elemzem az eredményeket.
Az irodalomban több lehetséges megoldó módszerről esik szó, gyártási probléma ütemezését elvégezték már többféle technika segítségével. A leggyakoribbak a MILP megoldó módszerek, melyek vegyes-egész lineáris programozáson alapulnak, de az S-gráf alapú megoldások is jellemzőek, valamint Petri-hálóval is oldottak már meg hasonló ütemezési feladatot. Ezekről az eredményekről szó esik a kapcsolódó irodalomban is, amelyeket fel tudunk használni arra, hogy az automatával elvégzett optimalizálás eredményeivel összehasonlítsuk őket.
Időzített automatával még nem járták körül bővebben a problémát, de érdemes vele foglalkozni, mert más problémaosztályok vizsgálata során hatékony módszernek bizonyult.
A feladat egy irodalmi példa, amelyben a meghatározott számú feladatot a gépek egy lépésben végzik el. A feladat célja, hogy minél több feladatot elvégezzenek a gépek, és minél kevesebb legyen a kész termékek tárolási költsége.
Dolgozatom második fejezetében az irodalmi háttérrel foglalkozom, a harmadik fejezetben részletezem a problémát, majd az időzített automatákról, a használt szoftverről és az LPTA alapú ütemezésről teszek említést. A negyedik fejezetben a teszteredményekről lesz szó, amelyeket korábbi megoldásokkal hasonlítok össze. A végén található az összefoglalás az ütemezés eredménye alapján, majd a dolgozat végére kerülnek a hivatkozások és a függelék.

2. fejezet

Irodalmi áttekintés

Bármely gyártási ütemezési feladat alapját ugyanazok a megadott paraméterek adják, bármilyen módszert használunk a megoldáshoz. Közös bennük, hogy adottak a feladatok, a feladatokat elvégezni képes berendezések vagy eszközök, egy adott időintervallum minden feladathoz, a cél pedig, hogy a kijelölt feltételek mellett kielégítsük a meghatározott végcélt. Az egyes feladatok intervalluma az az időtartam, ami a munka elvégzéséhez szükséges, ez általában minden esetben egyedien meghatározott, tehát minden különböző feladatnak eltérő mennyiségű időre van szüksége. A feladatoknak lehet elvégzési határideje is, amit szintén figyelembe kell venni az ütemezés folyamán. Az irodalomban a feladatokat task megnevezéssel is használják, a munkát elvégző gépeket unit-ként, a termékeket pedig product-ként.
A problémákat különböző szempontok alapján lehet kategorizálni, az egyik csoportosítás alapján megkülönböztetünk sztochaikus és determinisztikus ütemezési feladatokat, ahol a sztochaikus típus azokat a problémákat jelöli, ahol a paraméterek futás közben kapnak értéket. A determinisztikus esetében az értékek előre be vannak állítva. A két csoportot viszont nem lehet élesen elkülöníteni egymástól, mivel lehetnek olyan estetek is, ahol egyes paramétereknek muszáj még futtatás előtt értéket adni, míg a többi tényező futás közben veszi fel őket. Szintén két csoportba oszthatóak a problémák a következő szempontból, ha az ütemezés nem elégít ki legalább egy korlátozást, akkor az nem megoldható (infeasible), minden egyéb esetben megoldható, tehát feasible.
Az gyártási ütemezési feladatok egyik altípusa az egylépéses ütemezési probléma (single stage probléma), ahol a feladatokon egy tevékenységet kell végrehajtani, hogy azt befejezetté lehessen nyilvánítani. Az egylépéses problémák mellett vannak más gyakori típusok is, például a simple multiproduct, ahol a feladatot lineárisan több lépésben kell elvégezni, a general multiproduct, ami a simple multiproduct-hoz hasonló, de ki lehet hagyni lépéseket. A multipurpose (többcélú) problématípusban a lépéseknek nincs meghatározott sorrendje, tetszőlegesen hajthatóak végre. Megkülönböztethetjük a precedens típust, amely hasonló a többcélúhoz, de nem feltétlenül lineárisan hajtódnak végre a feladatok, és a general network típust, ahol a feladatokat az inputjaik és outputjaik

Termékek	Feldolgozási idő (óra)			Határidő
	u1	u2	u3	
P1	2,5	1,75	-	15
P2	-	0,6	2,84	20
P3	1	1,23	2	18
P4	0,57	0,98	-	16
P5	1,65	2,3	3	21
Beállítás	0,1	0,05	0,2	

2.1. ábra. Példa egylépéses ütemezési feladat megadására.

alapján különböztetnek meg.
A single stage problémáknak a megoldásához néhány paraméternek adottnak kell lennie, például a gépek számának, valamint annak, hogy ezek a gépek azonosak-e. Két gép akkor tekinthető azonosnak, ha ugyanazokat a munkákat képesek elvégezni ugyanannyi idő alatt. Szükség van továbbá a feladatok számára és típusára. A gépek az alábbiakban felsorolt típusúak lehetnek.

- **1 - Single Machine:** Egy gép elérhető, amelyen minden feladatot végre lehet hajtani. A termékeknek (feladatoknak) különböző feldolgozási ideje van.
- **Pm - Identical paralell machines**(Azonos párhuzamos gépek): m számú azonos gép áll rendelkezésre, amelyeken bármelyik egylépéses feladat végrehajtható.
- **Qm - Paralell machines with different speed** (Párhuzamos gépek különböző sebességgel): Hasonló az előző pontban említett típushoz, de minden gépnek meghatározott sebessége van.
- **Rm - Unrelated machines in paralell**(Párhuzamos független gépek): Hasonló a Single stage típushoz, de a feladatok elvégzési ideje egy-egy gépen inputként meghatározott.

2.1. Az egylépéses ütemezés ábrázolása

Az egylépéses ütemezési problémákat általában táblázat segítségével adják meg, ahol a sorokban tüntetik fel a munkákat, az oszlopokban pedig a rendelkezésre álló berendezéseket, a táblázatbeli metszéspontjaik ábrázolják az egyes munkák megfelelő berendezéseken való elvégzésének munkaidejét. Kopanos részletesen foglalkozott egylépéses ütemezéssel.

A fenti ábra példa egy ütemezési feladat táblázattal való megadására, ahol a feldolgozási idő mutatja a P-vel jelölt termékek egyes gépeken való munkaidejét, a gépek u-val jelöltek. Néhány feladatot nem tud minden gép végrehajtani, ezek nem rendelkeznek feldolgozási idővel az adott gépen.
Minden feladathoz fel van tüntetve a hozzá tartozó határidő, valamint a gépek beállási ideje, amely a bekapcsolás után szükséges időt mutatja, amíg a gép nem tudja elkezdeni a munkát.

2.2. Egyéb megoldó módszerek

Az egylépéses ütemezési feladatok témakörével még nem foglalkoztak részletesen az irodalomban, viszont vannak olyan módszerek, amelyekkel már a legtöbb problémaosztály szempontjából foglalkoztak. Az alábbi néhány eljárás a legnépszerűbb megoldók közé tartozik.

2.2.1. MILP modellek

A MILP modellek, tehát a vegyes-egész lineáris programozási modellek a legelterjedtebb megoldó módszerek közé tartoznak, és több altípusuk létezik.

Time discretization based - Időfelosztásos módszerek

Az időfelosztásos modellek előnye, hogy széles skálán mozog a megoldható problémák típusa. A módszer alapján időpontokat és időréseket különböztetünk meg. A időrés az az intervallum, amely egyik időponttól a következőig tart, az időpont pedig ennek a fordítottja, tehát az időrés kezdete az egyik időpont, az időrés vége pedig egy másik.
A feladatokhoz bináris változókat rendelnek aszerint, hogy a feladat az adott időpontban elvégzésre kerül, vagy nem. Amikor a feladat abban az időpontban megvalósul, akkor 1 lesz a bináris változó értéke, ha nem, akkor 0. Így annyi bináris változóra lesz szükség, ahány időpontot meghatároztunk. Lehetőleg minél kisebb számú időpont felvételével kell megtalálni az optimális megoldást a modell bonyolultságának csökkentése érdekében.

Két altípust különböztetnek meg az alapján, hogy az időpontokat az optimalizálás előtt meghatározzák, ezek a Fix időpontos időfelosztásos módszerek, vagy pedig csak a feladat közben kerül meghatározásra az időpontok száma. Utóbbiakat Variable time model-eknek hívják, és a lényegük, hogy minél kevesebb bináris változóra legyen szükség. A modellekben folyamatos változókat használnak, amik meghatározzák mindegyik időponthoz tartozó feladatokat.

2.2.2. S-gráf

Az első gráf alapú optimalizációra fejlesztett módszer, amely nemcsak vizuálisan szemlélteti a folyamatot, de egyben egy matematikai modell is. Irányított

gráfokból áll, amelynek a csomópontjai feladatok és az azokon elvégzendő műveletek, amelyeket az élek kötnek össze őket. Ezen kívül tartalmaznak ütemezési íveket (nyilakat), amelyek a meghozott ütemezési döntéseket modellezik.
Létezik ütemezési döntések nélküli S-gráf is, ezt recept gráfnak hívják.
A nyilak, amelyek a csomópontokat kötik össze, a függőségeket reprezentálják a következő esetekben:

2.2.3. Petri háló

A Petri hálót és az időzített automatát is gyakran alkalmazzák diszkrét esemény rendszerű ütemezési problémákhoz, és hogy kötegelt feladatok elvégzéséhez is alkalmas legyen, ki kellett egészíteni időzítéssel ezeket a módszereket. Hatékonyak, mivel jól szemléltetik a modellt, és a megfelelő felépítés mellett elkerülhetőek a hibák. Bár több előnyük is van a korábban említett népszerű megoldó módszerekhez képest, összességégben hatékonyságuk még elmarad a MILP és S-gráf alapú megoldó módszerekétől.
Az időzített Petri háló alapja, hogy az átviteli jel késleltetés (delay) alapján jön létre. Többen is foglalkoztak a témával, Ghaeli foglalkozott a kötegelt folyamatok ütemezésével ilyen módon, Soares pedig megpróbálta kiterjeszteni a modellt, és valós idejű ütemezést mutatott be kötegelt rendszerekre Petri háló segítségével.

3. fejezet

Problémadefiníció

A probléma egy egylépéses szakaszos eljárás ütemezéséhez kapcsolódik, ahol minden termék egy termelési lépés alatt készül el, ezeket a hívjuk munkáknak. A munkákat bármelyik gép (unit) elvégezheti, de egy munka csak egy géphez rendelhető hozzá. Ugyanígy egy gép egyszerre csak egy feladaton dolgozhat, és ha már egy munkát elkezdett, azt egy másik nem előzheti meg.
Adott a munkák és a gépek száma, valamint a gépeknek van egy meghatározott üzembe állási ideje, ez mindegyik berendezés egyedi tulajdonsága. Két munka elvégzése között felszámolunk átállási időt, amíg a gép testre szabja saját beállításait a következő feladathoz. A gyakorlatban ez tisztítást, újra beállítást és egyéb karbantartást jelent. Az átállási időt kétféleképpen lehet megadni, az egyik típus a szekvenciafüggő, amikor a feladatok sorrendje határozza meg az értéket. Mennyiségét az szabja meg, hogy az előző és az utána következő munka között mennyi időre van szüksége a berendezésnek. A másik megadási mód a szekvenciafüggetlen típus, amikor csak a berendezéstől függ az átállási idő. Mindkét modellt be lehet állítani úgy, hogy minkét típus függjön a géptől és a feladattól is. Itt...
A munkákat minden gép különböző idő alatt tudja elvégezni, de olyan eset is lehet, amikor egy gép nem tudja elvégezni az adott munkát. Minden feladat rendelkezik határidővel, amit nem léphet át, miközben várakoznia kell, ha a határidő előtt elkészül. Az ütemezés célja, hogy minimalizáljuk a várakozás költségeit, emellett viszont előfordulhat, hogy a megoldás nem elégít ki a korlátozásokat, így infeasble lesz. Ha van feasible megoldás, szeretnénk lehetőleg az összes munkát elvégezni határidőre, valamint a modellt kiegészíteni korlátozásokkal úgy, hogy minél kevesebb idő alatt elvégezze az ütemezést, és minél optimálisabb eredményt adjon.
A feladatban munkákat és gépeket különböztetünk meg, amelyeket később P-vel és U-val jelölünk, a product és unit szakirodalomban használt megnevezések után.

4. fejezet

Időzített automaták

4.1. Időzített automaták

A később bemutatásra kerülő problémát időzített automata segítségével oldottuk meg. Egy automata eseményekből és állapotokból áll, az időzített automata pedig kiegészül órákkal, amelyek mérik a globális időt, vagy egy konkrét automata idejét.
Az időzített automata képlete

$$(K, \Sigma, C, Tra, Inv, s)$$

ahol

K az állapotok halmaza
Σ az események halmaza
C az órák halmaza
Tra K x ϕ(C) x Σ x C x K időzített transitions
Inv K $\rightarrow$ ϕ(C) state invariants -
s a kezdőállapot

Az időzítést egy egyszerű szó levezetésével tudjuk bemutatni, ahol az események meghatározott időben történnek.
Példának vesszük az *ababb* szót.

(a,1) (b,3) (a,4) (b,6) (b,10)

A fenti képlet azt mutatja, hogy melyik időpillanatban történhet az esemény, amiből kiszámíthatjuk, hogy a betűk mennyi késleltetéssel követik egymást.

d1$\rightarrow$ **a** $\rightarrow$d2$\rightarrow$ **b** $\rightarrow$d1$\rightarrow$ **a** $\rightarrow$d2$\rightarrow$ **b** $\rightarrow$b4$\rightarrow$ **b**

ahol d a késleltetést (delay-t) mutatja.
Az első a 1 delay eltelte után kezdődhet el, és mivel b 3 delay után következik, a-t követően 2 delay-t kell várnia, a következő betűk pedig ennek alapján

ugyanezt a szabályt követik.

HIVATKOZÁS!!!!

4.2. UPPAAL Cora

A választott irodalmi példa időzített automatákkal való ütemezését az UPPAAL Cora nevű szoftver segítségével modelleztük. A szoftver alkalmas az automaták modellezésére, ütemezésére és optimalizálására meghatározott paraméterek alapján.
Az UPPAAL segítségével sablonokat (template-eket) hozhatunk létre, ahol minden egyes template egy különböző automata modellezésére szolgál. A sablon állapotokat és éleket tartalmaz, ahol az élek az állapotátmenetet szimbolizálják. Az állapotokhoz megadhatunk nevet, illetve valamilyen korlátozást, ezen kívül beállíthatjuk az állapotát, ami initial, urgent vagy committed lehet. Az initial az automata kezdőállapotaként jelöli meg a kijelölt státuszt, a committed még inkább korlátozó, mint az urgent, tehát nem késleltetheti a következő átmenetet. Ugyanitt beállítható az is, ha az adott állapotban növekszik a költség valamely egyéb korlátozásból adódó várakozás miatt. Ekkor az állapot beállításaiban az Invariant pontban adható meg az idő előrehaladásával számított költség mértéke.
Az irányított élek mutatják, hogy melyik állapotból melyikbe van lehetőség átkerülni, ezen kívül pedig egyéb beállítások is megadhatóak. Minden élre megadható Select, Guars, Sync és Update információ. A Select-ben értéket lehet adni változóknak, a Guard korlátozást állít be, aminek teljesülnie kell, hogy a feladat a következő állapotba kerüljön. A Sync lehetőséggel különböző automaták állapotátmeneteit lehet összehangolni, ehhez csatorna létrehozására van szükség. Az Update segítségével frissíthetőek a változók értékei.
Az automata template-ek az Editor menüpont alatt helyezkednek el, és itt található még a Declarations menüpont is.
A Declarations pontban az egész rendszerre vonatkozó változókat és értékeket lehet megadni, valamint függvényeket létrehozni. Az értékeket egészként érdemes meghatározni, mert a szoftver nem számol lebegőpontos alakban. Definiálhatunk órákat (clock), illetve csatornákat (chan), ezek az automaták közötti szinkronizálást segítik elő.
Ugyanitt lehet megadni az automaták számát abban az esetben, ha több hasonlóra is szükség van, így nem kell külön-külön létrehozni őket más paraméterekkel. Minden template-hez definiálhatunk lokális változókat és paramétereket, ahol a paraméterek segítségével például meg tudjuk különböztetni a példányokat, ha a modell példányosítva van. Ekkor hozzárendelünk egy ID-t, amely sorszámot ad az automatáknak. A lokális változók között gyakran definiálunk órát, ha például egy elvégzendő feladatról van szó, külön mérhessük a munkaidejét, amit ilyenkor a modell elején le is kell nullázni. Az Editor lapon található még a System Declarations pont, ami az automaták konkrét példányosítását végzi.
Az Editor mellett két fontos menüpont található. Ha nem vétettünk szintaktikai

hibát, a szimulátor betölti a létrehozott automaták összes példányát, mellette pedig megjeleníti a változókat, amelyek először a kiinduló állapotban vannak, ahogy az automaták is. Itt lehetőség van random megoldást lefuttatni, ekkor szinte biztos, hogy nem a legoptimálisabb eredményt kapjuk. A lépéseket saját magunk is kiválaszthatjuk, a végeredményt pedig mindkét esetben vissza lehet játszani, vagy elmenteni egy külön fájlba.
A szimulátor közben bemutatja, hogyan változtak az értékek, hogy az automaták melyik állapotukban vannak, valamint egy másik ábrán szekvencia diagramon láthatjuk a szinkronizáció lépéseit, és az automaták állapotátmenetét.
A harmadik menüpont a Verifier, itt a Query-ben meg lehet adni korlátozást, amely alapján az UPPAAL lefuttatja az ütemezést, majd kiírja az eredményt, amely két fajta lehet. Ha a beírt korlát alapján talált megoldást, akkor megkapjuk a *Property is satisfied* üzenetet zölddel kiírva a Status pont alatt. Az Overview-nál is megjelenik a megadott Query, mellette pedig egy zöld jel. Ellenkező esetben a *Property is not satisfied* üzenetet kapjuk, az Overview pedig piros jelet tesz a Query-ben megadott korlát mellé.
A főmenüben leginkább az általános lehetőségeket találjuk, a Tools és az Options pontokban találhatóak egyéb beállítások az automaták ütemezésével kapcsolatban. Leellenőrizhető a Declarations és a template-ek definiálása során megadott adatok és korlátozások szintaktikai helyessége, az Options pedig az ütemezési beállításokat tartalmazza.
Átállítható például, hogy ütemezés során mélységi, szélességi vagy egyéb keresést alkalmazzon, valamint hasznos funkció, hogy ha a Diagnostic Trace értékét some-ra állítjuk, akkor abban az esetben, ha a Query-ben definiált lekérdezés teljesül, a megoldást be tudja tölteni a szimulátorba. A szimulátorban grafikusan, diagramon és változónként is elemezhető az eredmény.

5. fejezet

LPTA alapú single stage ütemezés

5.1. Leírás

5.2. Template

5.3. query

6. fejezet

Teszteredmények, összehasonlítás

A modellt először a szükséges korlátozásokon kívül egyéb korlátozásokkal nem egészítettük ki, amikkel csökkenteni lehetett volna az ütemezés futási idejét. A Query-ben megadott cél az volt, hogy olyan megoldást találjon, amikor az összes feladatot sikeresen elvégzi, tehát leszállításra kerülnek határidőre.
Mivel az modellt 10 percen belül nem tudta lefuttatni az UPPAAL, az irodalmi példában szereplő 20 feladat helyett először 1 feladattal került tesztelésre az automata futási ideje, majd egyesével növeltük a feladatok számát addig, amíg a modell képes volt tíz percen belül megoldást találni rá. Az eredményeket az alábbi táblázat tartalmazza.

Az eredmények alapján megállapítható, hogy korlátozás nélkül a 20 munka közül csak 5 munkával futott le az ütemezés tíz percen belül, a hatodik munka hozzáadása után tíz perc alatt nem kaptunk eredményt. Emiatt a további bővítést már nem végeztük el, beláthatóan egyiknek sem lett volna eredménye

Munkák száma	Futási idő	Munkák száma	Futási idő
1 munka	0,08 sec	11 munka	-
2 munka	0,11 sec	12 munka	-
3 munka	0,36 sec	13 munka	-
4 munka	1,99 sec	14 munka	-
5 munka	36,5 sec	15 munka	-
6 munka	10+ min	16 munka	-
7 munka	-	17 munka	-
8 munka	-	18 munka	-
9 munka	-	19 munka	-
10 munka	-	20 munka	-

6.1. ábra. Futási eredmények korlátozás nélkül.

belátható időn belül.

Hivatkozasok

bibtex

A. függelék

Jelolesek