

TDK-DOLGOZAT

Éles András, Kovács Balázs, Tóth Benjámín



SZAKASZOS ÜZEMŰ TERMELŐ RENDSZEREK ÜTEMEZÉSE: MEGOLDÓ MÓDSZEREK EMPIRIKUS ANALÍZISE

Éles András Mérnök informatikus BSc, 5. félév	Kovács Balázs Mérnök informatikus BSc, 5. félév	Tóth Benjámín Mérnök informatikus BSc, 5. félév
Pannon Egyetem Műszaki Informatikai Kar	Pannon Egyetem Műszaki Informatikai Kar	Pannon Egyetem Műszaki Informatikai Kar

Témavezető:
Hegyháti Máté
PhD hallgató, PE MIK

Absztrakt

Az egyre összetettebb gyártási folyamatok miatt napjaink termelő rendszereinek hatékonysága nagymértékben függ a berendezések üzemeltetésének módjától. Ennek is köszönhető, hogy az elmúlt két évtizedben a szakaszos gyártási folyamatok ütemezése fontos területté vált.

Rövidtávú ütemezési feladatok megoldására a nagy számítási igény és a feladatok sokszínűsége miatt gyökeresen eltérő modellezési technikákat fejlesztettek ki. Az irodalomban gyakran alkalmaznak kevert egész értékű lineáris programozási (MILP) modelleket, de ezek mellett léteznek kombinatorikai megközelítések is.

Munkánk során létrehoztunk egy keretrendszert, melyben implementáltuk a különböző technikákat használó MILP modelleket, és az irodalomban szereplő ütemezési feladatokat egy általunk létrehozott általános formátumban. Dolgozatunk fő eredménye irodalomból ismert ütemezési feladatokon elvégzett széleskörű teszteken alapul. Az empirikus vizsgálatokból következtetéseket vontunk le, hogy az ütemezési feladatok tulajdonságai alapján az irodalomban jelenleg legnépszerűbb módszerek közül melyek használata a legkedvezőbb.

Tartalomjegyzék

1. Bevezetés	8
2. Szakaszos üzemű rendszerek ütemezési feladatai	10
2.1. A recept felépítése	11
2.2. Feladatok osztályozása gyártórendszerek típusa szerint	12
2.3. Feladatok osztályozása tárolási stratégia szerint	15
3. Ütemezési módszertanok	18
3.1. Matematikai programozási modellek	18
3.1.1. MILP modellek általában	18
3.1.2. Időalapú modellek	20
3.1.3. Precedencián alapuló modellek	25
3.1.4. Slot alapú modellek	27
3.1.5. Egyéb megközelítések	29
3.2. Kombinatorikai megközelítések	29
3.2.1. S-gráf módszertan	29
3.2.2. Időzített állapotú Petri hálók	31
3.2.3. Időzített automaták	32
4. Egységes formátum ütemezéshez	35
4.1. Felhasznált nyelv	35
4.2. Feladatok egységes formátuma	36
4.3. Modellek implementációja	36
5. Korábbi tesztek	39
6. Empirikus vizsgálatok	42
6.1. Modell adatbázis	42
6.2. Tesztkörülmények, módszerek	42
6.2.1. Esettanulmányok	45
6.3. Esettanulmány I.	46

6.4. Esettanulmány II.	47
6.5. Esettanulmány III.	48
6.6. Esettanulmány IV.	49
7. Összefoglalás	63
A. Futási eredmények	64

Ábrák jegyzéke

2.1.	Gantt-diagram a single-stage feladathoz	12
2.2.	Blokkdiagram a multiproduct feladathoz	13
2.3.	Blokkdiagram egy multipurpose feladathoz	14
2.4.	STN az általános célú recept által leírt feladathoz	15
2.5.	Receptgráf a multipurpose feladathoz	16
3.1.	Diszkrét időintervallumok (STN modell). A téglalapok az általánosan hálózat topológiájú folyamat taszkjait jelölik.	21
3.2.	Globális időpontok (STN modell). A téglalapok az általánosan hálózat topológiájú folyamat taszkjait jelölik.	22
3.3.	Berendezés-specifikus eseménypontok. Az egyes események berendezésként eltérő időpontokban is történhetnek.	23
3.4.	Berendezés-specifikus közvetlen precedencia. Minden berendezés taszkjai között előidejűségi sorrend áll fenn.	26
3.5.	Közvetlen precedencia. Szétválik az előidejűség és a taszkok berendezésekhez rendelése.	27
3.6.	Általános precedencia. Minden közvetett, vagy közvetlen előidejűséget bináris változó jelez.	28
3.7.	Slot alapú modellek. A berendezések időszeleteinek és a multipurpose folyamat taszkjainak a megfeleltetése történik.	28
3.8.	Példa S-gráfra. A termékek és a berendezések taszk-sorrendjét is súlyozott, irányított élek jelzik.	30
3.9.	Példa Petri hálóra. A tranzíciók által kell elérni, hogy a termékek végállapotaiba tokenek kerüljenek.	31
3.10.	Példa időzített automatára. Az egyes berendezések (és az ábrán be nem rajzolt erőforrások) határozzák meg az automata állapotainak „dimenzióit”.	33

4.1.	Wrapper - az egységes formátumot kezelő rendszer. Segítségével egy feladat adatfájlja tetszőleges, a megoldására képes módszer modelfájljával együtt fordítható egy segédprogrammal, automatizálva ezzel a tesztelést. A nyilak jelölik az adatok lehetséges áramlási útját	37
4.2.	QR kód a wrapper forrásához	38
5.1.	Makespan eredmények nagy tesztesetekre	41
6.1.	Az adatbázisban lévő modellek listázása	43
6.2.	A webalkalmazás elérhetősége	43

Táblázatok jegyzéke

6.1. Adatok az I. esettanulmányhoz - berendezések és végrehajtási idők.	46
6.2. Adatok a II. esettanulmányhoz - berendezések és végrehajtási idők.	47
6.3. Adatok a III. esettanulmányhoz - berendezések és végrehajtási idők.	48
6.4. Adatok a IV. esettanulmányhoz. Választási lehetőségek szintenként, berendezések és végrehajtási idők.	49
6.5. A kis feladat futási adatai	50
6.6. A voudouris NIS feladat futási adatai	51
6.7. A voudoris UIS feladat futási adatai	54
6.8. A ferrer NIS feladat futási adatai	58
6.9. A ferrer feladat futási eredményei	59
6.10. A ferrer UIS feladat futási eredményei	60
6.11. A ferrer UIS feladat futási eredményei	61
6.12. A festékipari feladat futási eredményei	61
6.13. A festékipari feladat futási eredményei	62
6.14. A festékipari feladat futási eredményei	62
A.1. Mendez2003 modell eredmények	65
A.2. Castro2004 modell eredmények 2.	66
A.3. Castro2004 modell eredmények 3.	67

1. fejezet

Bevezetés

Napjainkban a gyártási folyamatok egyre összetettebbé válnak, ennek következményeként a termelő rendszerek hatékonysága egyre nagyobb mértékben függ a berendezések megfelelő üzemeltetésétől. A termelő rendszerek ütemezése ennek megfelelően fontos területté vált. Ipari méretű feladatok megoldása komoly nehézséggel járhat, mivel a rendszer növekedésének hatására a modell komplexitása exponenciális mértékben nő. Ebből következik, hogy sokszor nehéz belátható időn belül biztosítani az optimalitás megoldást.

Dolgozatunk 2. fejezetében először ismertetjük a szakaszos üzemű rendszerek működését. Ezek a folyamatok nagyfokú változatosságot mutatnak, ezért is van szükség eltérő módszerekre ütemezésükhöz. Bemutatásuk során főként az ütemezésük szempontjából fontos fogalmakra, tulajdonságokra koncentrálnak. Számos, a témakörben használt lehetőséget ismertetünk a feladat leírására, illetve az eredmény szemléltetésére.

A fejezetben leírjuk az általunk alkalmazott kategorizálási szempontokat, melyek szerint számos irodalmi példát megvizsgáltunk. Ennek eredményeként, részben munkánk segítségével, a kategorizált feladatokat adatbázisba gyűjtöttük, mely egy saját fejlesztésű webalkalmazás segítségével kezelhető.

A 3. fejezetben bemutatjuk az irodalomban jelenleg népszerű módszertanokat. A feladatok sokszínűsége és nehézsége miatt nagy eltéréseket találunk a probléma modellezésében, felhasználhatóságban és hatékonyságban. A különböző, matematikai programozási modellekre épülő módszerek a feladatok széles körére alkalmazhatóak, azonban más módszertanok rendelkezhetnek olyan tulajdonságokkal, melyek bizonyos típusú példánál nagy előnyt jelenthetnek. Az S-gráf módszertan az ütemezési feladatok egyedi szerkezetét kihasználva lényegesen gyorsabb lehet, illetve mentes az intuitív hibáktól. Ezek mellett bemutatjuk a Petri-hálókat, és az árazott időzített

automatákat is.

Az 4. fejezetben bemutatjuk az általunk létrehozott keretrendszert, melyben implementáltunk számos lineáris programozási modellt, illetve irodalomban megjelent ütemezési feladatokat, egy általunk létrehozott egységes formátumban.

Dolgozatunk fő eredménye irodalmi példákon elvégzett széleskörű teszteken alapul. Munkánk korábbi fázisát az 5. fejezetben tárgyaljuk, majd a 6. fejezetben megvizsgáljuk a különböző megközelítések hatékonyságát.

2. fejezet

Szakaszos üzemű rendszerek ütemezési feladatai

Napjaink vegyipari gyártási folyamatai két csoportra bonthatóak: vannak folyamatos és szakaszos üzemű rendszerek. A folyamatos üzemű rendszerek lényege, hogy a gyártás folyamán az alapanyag időben folyamatosan áramlik be a rendszerbe, és a termék folyamatosan áramlik ki a gyártórendszer végén. Szakaszos rendszereknél egy lépés során a berendezésbe először beáramlik az összes alapanyag, majd végrehajtódik a feladat, végül a végtermék kikerül a berendezésből. Egy szakaszos rendszerű gyártási folyamat ilyen, tovább nem bontható lépések, más szóval taszkok sorozatából áll. Mindkét rendszernek megvannak az előnyei és a hátrányai.

Adott termékek gyártásához a folyamatos rendszerek lényegesen olcsóbbak az ugyanerre a célra kialakított szakaszos rendszereknél, azonban ha változtatni szeretnénk az előállítandó terméken, akkor előfordulhat, hogy akár az egész gyártósort le kell cserélni. Ezzel szemben szakaszos rendszereknél, mivel a gyártás általában több berendezésen keresztül történik, könnyebb átállni másik termék előállítására, mivel csak a rendszer részeit szükséges módosítani. Munkánkban csak szakaszos rendszerekkel foglalkozunk.

A szakaszos üzemű rendszereknek nagyon sokféle típusa létezik. Ezek különböznek egymástól a gyártandó termékek számában, a használható berendezések típusában, mennyiségében, a felhasználandó anyagok tárolhatóságában és még sok egyéb paraméterben. Az ütemezés célja megadni, hogy melyik berendezés, mikor, melyik gyártási folyamatot végezze. Ennek szemléltetésére a gyakorlatban egyik legelterjedtebb eszköze a Gantt-diagram (2.1. ábra), melynek függőleges tengelyén a berendezések, vízszintes tengelyén az idő helyezkedik el. Az egy termékhez tartozó folyamatokat azonos színnel, vagy számmal jelölik.

Az ütemezési feladatok megoldása során valamely szempont szerinti op-

timális ütemezést keresünk. Példák ütemezési célokra:

- Adott mennyiségű termék legyártása minél rövidebb idő alatt (makespan minimization).
- Adott idő alatt minél több termék legyártása (throughput maximization).
- Termékek legyártása határidőre (due date).
- Működési költségek minimalizálása (operation and changeover costs)

Tesztjeinkben csak makespan minimalizálással foglalkozunk.

A szakaszos üzemű ütemezési feladatokról több áttekintés is megtalálható az irodalomban ([1]).

2.1. A recept felépítése

Egy szakaszos rendszerű ütemezési feladatot legjobban az úgynevezett recept adja meg. A recept azokat a minimális információkat tartalmazza, amelyek egy termék előállításához szükségesek. Az ISA SP88 szabvány a recepteknek négy szintjét definiálja, melyek használata a felhasználás szintjétől függ. A szintek a következők [2][3]:

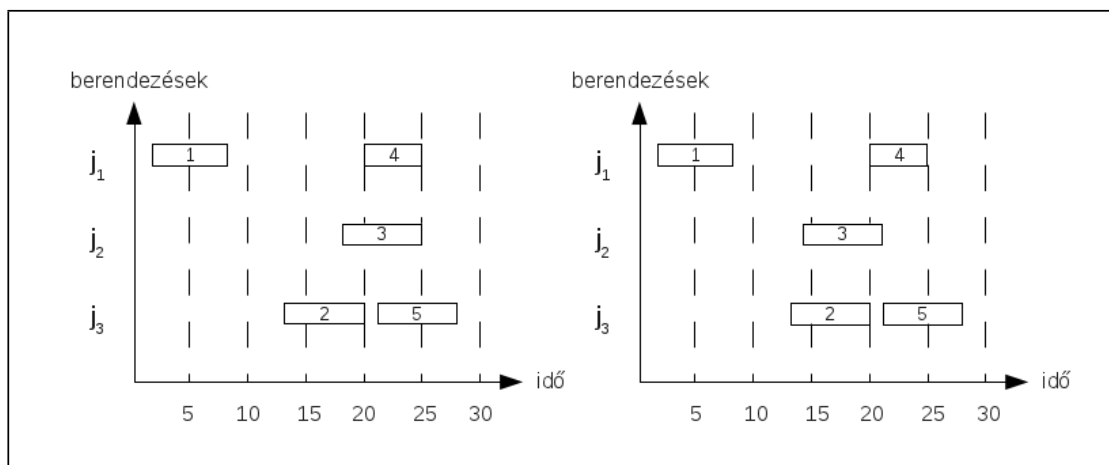
- Az általános recept azonosítja a nyersanyagokat, a relatív mennyiségüket, és az előállítandó anyagmennyiséget. Ez a recept nem tartalmaz berendezés és hely specifikus információkat.
- A hely recept egy, a hely specifikus információkkal kibővített általános recept.
- A mester recept berendezés specifikus információkat tartalmaz, tartalmazza a nyersanyagokat a rendelkezésre álló mennyiségekkel, valamint a termékek előállításának folyamatát.
- A kontroll recept a mester recepthez képest tartalmaz további információkat azon berendezésekről, melyeket egy termék egy batch-ének előállítására használnak (batch: egy termékből egy gyártási folyamat alatt legyártható mennyiség). Ez a recept akkor jön létre, mikor egy batch-et beütemeznek egy berendezéshez.

Az elkövetkezendőkben a mester receptet használjuk, amit a továbbiakban receptnek hívunk.

2.2. Feladatok osztályozása gyártórendszerek típusa szerint

Aszerint, hogy egy termék előállításához milyen taszkok végrehajtását igényli, és azokat milyen sorrendben kell végrehajtani, öt lényegesen különböző feladattípust különböztetünk meg. Ezek egymásba ágyazódnak, a speciálistól az általános felé haladva mutatjuk be őket.

2.1. ábra. Gantt-diagram a single-stage feladathoz



Single-stage feladatok

Az ütemezési feladat single-stage, ha egy rendelkezésre álló nyersanyagból egyetlen taszk végrehajtásával megkapunk egy terméket. Minden terméket csak egy taszk végrehajtásával állíthatunk elő, amire néhány előre meghatározott berendezés valamelyikében kerülhet sor.

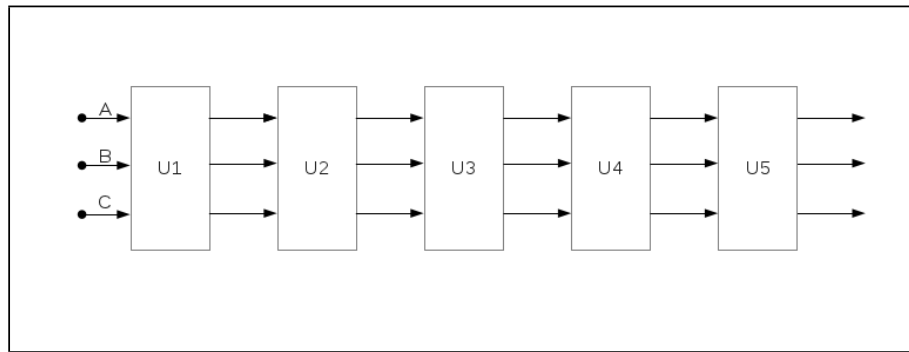
Multiproduct feladatok

Multiproduct feladatoknál a gyártórendszer több egymásra épülő szintből áll (2.2. ábra). Minden szintnek saját berendezéskészlete van, és mindegyik berendezés pontosan egy szinthez tartozik. Egy-egy nyersanyagnak végig kell mennie a szinteken úgy, hogy minden szint berendezéseiben legfeljebb egy taszkon esik át. Ilyenkor egy taszk vagy egy nyersanyagot, vagy egy korábbi

szint által létrehozott köztes terméket használ fel, és egy végterméket, vagy egy későbbi szint által igényelt köztes terméket állít elő. Egy multiproduct feladatban több termék is lehet, de mindegyik a fent leírt szinteken megy át ugyanabban a sorrendben, de megengedett, hogy egy szint kimaradjon egy termék receptjéből.

Jegyezzük meg, hogy egy multiproduct feladat több lépcsős abban az értelemben, hogy minden szintje külön-külön egy single-stage feladatnak felel meg.

2.2. ábra. Blokkdiagram a multiproduct feladathoz

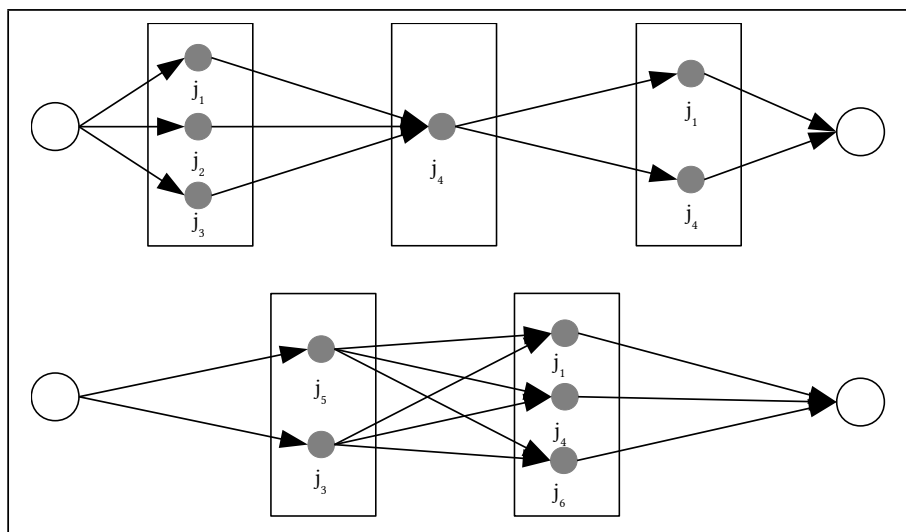


Multipurpose feladatok

Multipurpose feladatról beszélünk, ha minden terméknek több, csak arra a termékre jellemző szinten kell átesnie (2.3. ábra). Minden termék minden szintje a gyártórendszer berendezéseinek egy meghatározott részhalmaza, melyek között lehetnek átfedések. Egy multipurpose feladat egy-egy terméke egy olyan feladat, amely abban különbözik az előző feladattípustól, hogy egy berendezés több szinthez is tartozhat.

Az irodalomban gyakran az általunk imént leírtnál speciálisabb feladattípust neveznek multipurpose feladatnak. Ezt a típust az jellemzi, hogy a multiproduct feladathoz hasonlóan a termékekhez tartozó szintek ugyanazok, csak a végrehajtási sorrendjük különböző. Mi azért használjuk az általánosabb multipurpose fogalmat, mert az irodalomban előforduló, multipurpose feladatokhoz írt modellek valójában ezt a bővebb feladattípust képesek megoldani.

2.3. ábra. Blokkdiagram egy multipurpose feladathoz



Precedential feladatok

Egy feladat akkor precedenciális, ha minden termék receptje megadható taszokok egy rögzített halmazával, melyeket valamilyen előidejűségi korlátozásokot (precedencia-relációkat) betartva kell végrehajtani. Egy taszkra olyan jellegű kikötések létezhetnek, hogy bizonyos más taszokok végrehajtása után kell elkezdeni, vagy más taszokok kezdése előtt kell befejezni. Minden taszk berendezések egy rögzített halmazával hajtható végre.

Ez a definíció magába foglalja a multipurpose feladatokat is, ahol termékenként taszokok egy adott halmazát kell végrehajtani fix sorrendben. Ez a legbővebb feladattípus, ahol egy termék legyártásához taszokok egy rögzített halmazának végrehajtása szükséges.

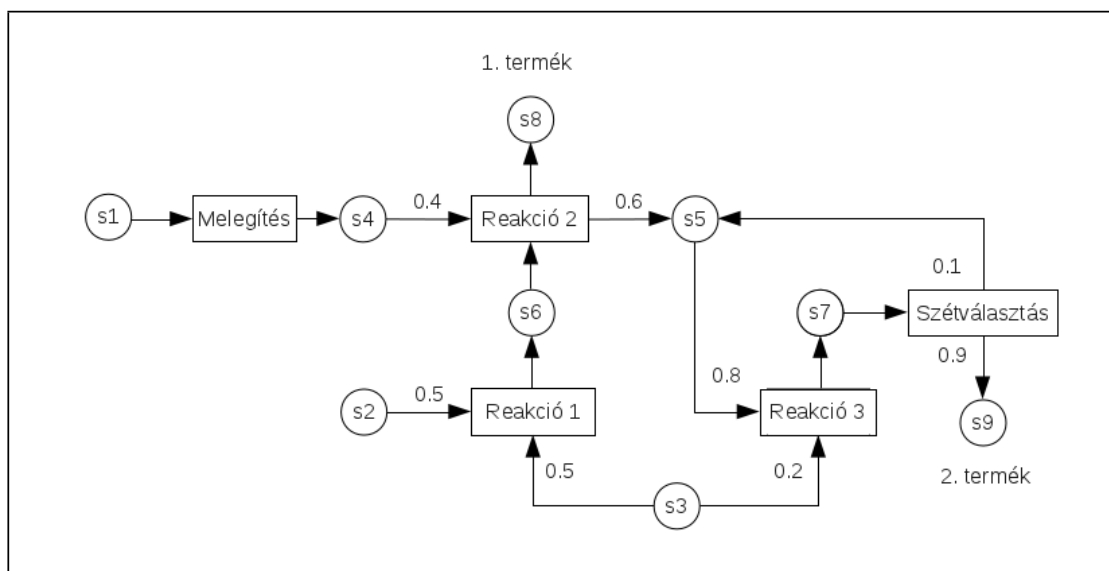
Egy precedenciális feladat leírható például az S-gráf reprezentáció segítségével [2].

Network feladatok

A network a legáltalánosabb feladattípus, amivel ütemezési feladatokat lehet leírni. Itt expliciten definiáltak a gyártási folyamat során létező anyagok. Adottak a taszokok, amelyek az anyagok adott arányú bemenetéből meghatározott arányú kimeneti termékeket állítanak elő. Nem rögzített, hogy mely taszkokat és hányszor hajtjuk végre. A recept hálózat topológiájú,

amely többféleképpen megadható. Az egyik lehetőség az STN (State Task Network [4]), amely egy, az anyagokból (state) és a taszkokból álló irányított páros gráf.

2.4. ábra. STN az általános célú recept által leírt feladathoz

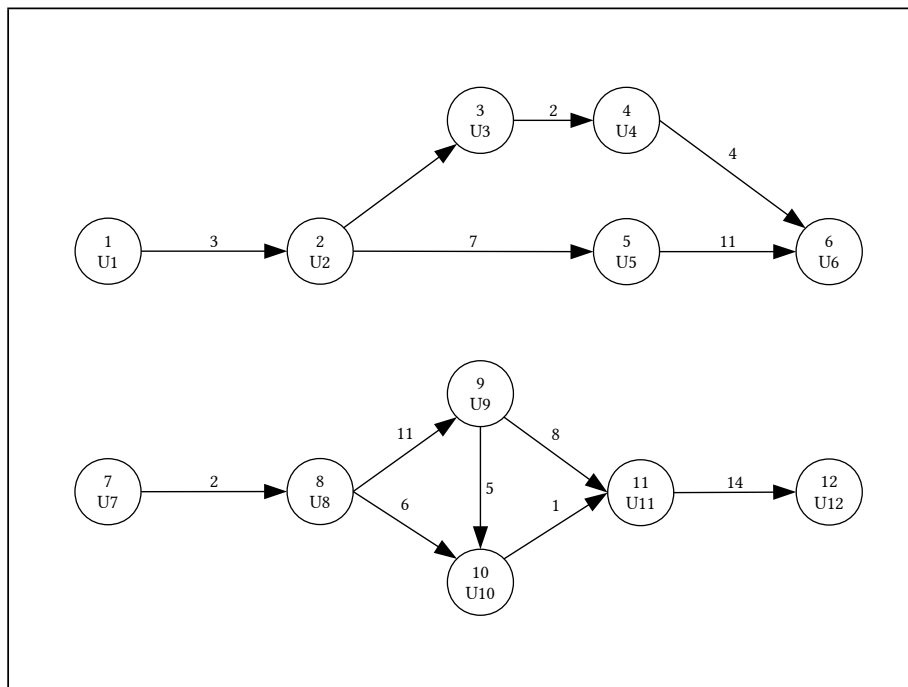


2.3. Feladatok osztályozása tárolási stratégia szerint

Nagyon fontos szempont a szakaszos üzemű rendszereknél, hogy a folyamatok között a köztes terméket hol, mennyi ideig, és továbbá mennyit lehet tárolni. Ezt egy feladatnál az úgynevezett tárolási stratégia határozza meg. Tárhely szerint elkülöníthető:

- UIS (Unlimited Intermediate Storage)
- FIS (Finite Intermediate Storage)
- CIS (Common Intermediate Storage)
- NIS (Non-Intermediate Storage)

2.5. ábra. Receptgráf a multipurpose feladathoz



- MIS (Mixed Intermediate Storage)

Az anyagok tárolási ideje szerint:

- UW (Unlimited Wait)
- LW (Limited Wait)
- ZW (Zero Wait)

UIS esetén a köztes anyagok tárolására korlátlan hely áll rendelkezésre. FIS tárolási stratégiánál anyagonként külön-külön véges hely van. CIS tárolási stratégiánál véges köztes tároló van az anyagok egy adott halmazához. NIS stratégiánál nincs tárolóhely a köztes anyagoknak, itt a berendezések funkcionálnak tárolóként a következő taszk elindulásáig. MIS stratégiáról akkor beszélünk, amikor különböző köztes anyagoknál különböző tárolási stratégia van jelen.

UW esetén a köztes anyag bármeddig tárolható. LW stratégiáról akkor beszélünk, ha a köztes anyag egy bizonyos idő eltelte után nem lesz felhasználható. A ZW a LW szélsőséges esete, amikor az köztes anyag az elkészülése után azonnal felhasználható.

Az, hogy milyen tárolási stratégiát használunk, befolyásolhatja a megoldások megvalósíthatóságát. A legfontosabb probléma, ami előfordulhat bizonyos stratégiáknál, az az úgynevezett cross-transfer. Több helyen bemutatták, hogy a UIS stratégiával megvalósítható, esetleg optimális megoldás nem biztos, hogy megvalósítható a többi stratégiával [5][6]. A probléma a következő: tekintsünk két berendezést, amelyek befejezték az aktuális folyamatot. A berendezések által előállított köztes termékeknek helyet kell cserélniük a gyártás folytatásához. Ennek megoldása köztes tároló nélkül nem megvalósítható, így UIS stratégiával gyakran jobb megvalósítható megoldás létezik, mint a többi tárolási stratégiával.

Mi jelen munkánk során csak UIS, NIS, UW tárolási stratégiákkal foglalkozunk.

3. fejezet

Ütemezési módszertanok

3.1. Matematikai programozási modellek

3.1.1. MILP modellek általában

Ütemezési feladatok széles skálája megoldható matematikai programozási modellek segítségével. A módszer lényege, hogy a döntések algebrai változók formájában jelennek meg. Ahhoz, hogy a változók értékei által reprezentált ütemezés megvalósítható legyen, bizonyos feltételeknek teljesülnie kell. Ezen feltételeket hívjuk korlátozásoknak, amelyek a modellben a változókra felírt matematikai összefüggések formájában fogalmazódnak meg. Egy ütemezés akkor és csak akkor megvalósítható, ha a változók értékei teljesítik a korlátozásokat. A megvalósítható ütemezések közül egy bizonyos szempont szerinti legjobbat keressük. Ez a szempont lesz a célfüggvény, amelynek értéke a változók valós függvénye. Az ütemezési feladat modellje tehát változók, korlátozások és egy célfüggvény együttese. A feladat a célfüggvény optimalizálása (maximalizálása vagy minimalizálása), feltéve, hogy a változók teljesítik a korlátozásokat.

Munkánkban csak vegyes-egészes lineáris programozási (Mixed Integer Linear Programming, MILP) modellekkel foglalkozunk. Ezek esetében a változók valós vagy egész értékűek (folytonosak vagy diszkrét), a korlátozások a változókra felírt lineáris egyenletek vagy egyenlőtlenségek, a célfüggvény pedig a változók lineáris függvénye. A továbbiakban egész és diszkrét változókon csak binárisakat (0-1 változókat) értünk.

Egy adott problémára felírt MILP modellben a célfüggvényt optimalizálni lehet általános célú megoldóval, ezek futási ideje viszont nagyban függ a modelltől. Egy MILP modell méretét, vagyis megoldásának nehézségét - gyakorlati tapasztalatok alapján - döntően az alábbi jellemzői határozzák meg:

- diszkrét változók száma - a szükséges idő általában exponenciálisan nő az egész (beleértve a bináris) változók számával;
- folytonos változók száma - több folytonos változó esetén lassabb a modell megoldása, bár a szükséges időre ez nincs olyan jelentős hatással, mint a diszkrét változók száma;
- korlátozások száma - több korlátozás kiértékelése több időt igényel;
- a modell LP-relaxációja - az elemi lépésnek tekinthető LP-relaxáció minél pontosabb (minimalizálásnál alsó, maximalizálásnál felső) becslést ad a vegyes-egészes feladat optimumára, annál hatékonyabb az egyes eseteket átvizsgáló algoritmus. Ha gyenge a becslés, akkor drasztikusan megemelkedhet az átvizsgált esetek száma. Ez a tulajdonság főként a Branch and Bound, (B&B) keretalgorithmussal való megoldásban jelentkezik.

Megjegyzendő, hogy ezen tulajdonságoknak a futási időre gyakorolt tényleges hatása a felhasznált szoftveres megoldótól és annak konfigurációjától függ. Az egész (bináris) változók számát nézve viszont általánosan a MILP problémák megoldása minden eddig ismert módszerrel exponenciális komplexitású. Ezért szempont minden matematikai programozási modellben a bináris változók számának elfogadható szinten tartása.

Egy ütemezési feladat többféleképpen modellezhető MILP modellel. A keresési tér alapesetben a megvalósítható ütemezések halmaza. Az egyszerűbb és gyorsabb megoldás érdekében azonban gyakran szükség van a keresési tér leszűkítésére. Ezzel az optimalitás általában sérül, a szűkebb problémára már csak a gyakorlatban szuboptimális megoldások érhetőek el, cserébe a probléma gyakorlatban kezelhetővé válik. A keresési tér megválasztásán túl a döntések, vagyis a modell változóinak kijelölése is jelentős hatással van a futási teljesítményre. Az alábbiakban ismertetjük a legfontosabb modellezési aspektusokat, és kategorizáljuk a létező módszereket.

Három ütemezéshez használt MILP modell típust különböztetünk meg, ezek a következők:

- Időalapú modellek
- Precedencián alapuló modellek
- Slot alapú modellek

Készültek a miénktől eltérő csoportosítást használó, részletes ismertető a létező ütemezési modellekről ([1]).

3.1.2. Időalapú modellek

Az időalapú modellekben a taszkoknak egy expliciten meghatározott időtengelyre illesztése történik. Az időhorizon gondolatban fel van osztva intervallumokra. Dönteni abban kell, hogy egy adott intervallumon (vagy intervallumok sorozatán) egy taszk végre legyen-e hajtva, vagy sem.

Általánosan elmondható az időalapú modellekről, hogy alkalmazhatóak a korábban említett legáltalánosabb feladattípuson, vagyis Network feladatok esetén, ami a precedencia alapú modellekre nem igaz. Időalapú modelleknél a legegyszerűbb a gyártási folyamat során felmerülő egyéb korlátozásokat utólag beiktatni a modellbe.

Jelentős hátrány, hogy az intervallumok számát a megoldás előtt rögzíteni kell. Ha alacsonyra választjuk az intervallumszámot, az idő előzőleges felosztása nagyban korlátozza a keresési teret, esetlegesen a modell nem is ad majd megoldást. A túl nagy intervallumszám viszont a modell méretét növelheti meg annyira, hogy az elfogadható időn belül le sem tud futni. Az intervallumszám optimális meghatározása nehéz, jelenleg nincs rá tökéletes módszer.

Többféleképp csoportosíthatóak az időalapú modellek, három általunk lényegesnek ítélt szempontot mutatunk be.

Elsődleges csoportosítási szempontként, az intervallumok definíciója szerint az alábbi modelltípusok léteznek:

- Diszkrét időintervallumos (Discrete Time Intervals);
- Globális időpontos (Global Time-Points);
- Berendezés-specifikus eseménypontos (Unit-specific Event-Points).

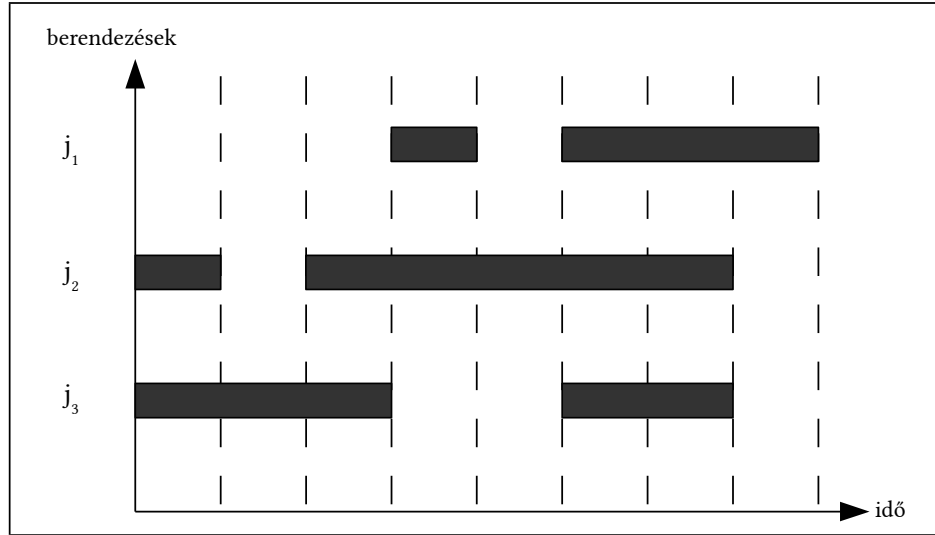
A berendezések reprezentációja szerint léteznek:

- STN (State Task Network) modellek;
- RTN (Resource Task Network) modellek.

A taszkok intervallumokhoz rendelésének (allokációjának) módja szerint kétféle modell létezik:

- "Start/stop allokációs" modell;
- "Delta allokációs" modell.

3.1. ábra. Diszkrét időintervallumok (STN modell). A téglalapok az általánosan hálózat topológiájú folyamat taszkjait jelölik.



Csoportosítás az intervallumok definíciója szerint

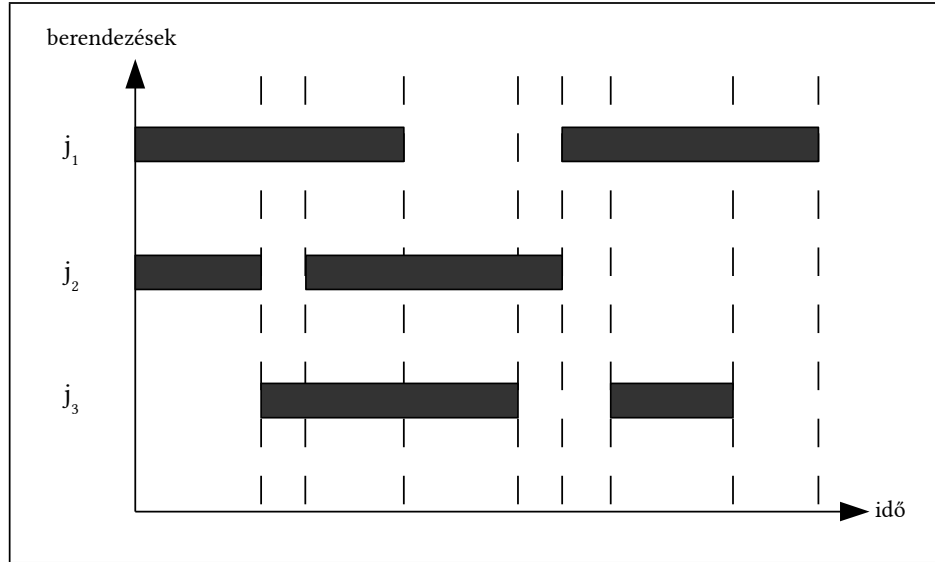
Diszkrét időintervallumos modellek. Egy lehetséges megoldás, hogy az időtengelyt azonos, előre meghatározott méretű intervallumokra osztjuk. Ez a diszkrét időintervallumos modellek esete ([4], [7]). A döntési változók a modellben azok, hogy egy adott taszk egy adott intervallumban (vagy intervallumok egymást követő sorozatában) fusson-e, egy adott berendezésben.

A módszer előnye, hogy az időpontok abszolút értelemben fixek, így minden olyan külső tényezőt implementálni lehet a modellbe, amelyek időtől függenek (pl. adott időpontbeli szállítmány, erőforráskiesés, vagy folyamatos megrendelések). Nagy hátrány, hogy a diszkrétizációval pontatlanságot veszítünk, továbbá nagy méretű feladatok esetén a bináris változók nagy száma miatt alkalmazhatatlan a modell.

Globális időpontos modellek. Ha az időtengelyt egységes, de meghatározatlan és eltérő hosszúságú intervallumokra van felosztva, a globális időpontok esetéről beszélünk. A döntési változók ugyanazok, mint a diszkrét időintervallumok esetén ([8],[9],[10]).

Előnye a modell típusnak, hogy kevesebb modellezési pontosságot veszít a diszkrétizáció hiánya miatt. Hátránya, hogy a modellben a globális időpontok alkalmazása az összes taszkra nézve továbbra is erős korlátozás, szűk

3.2. ábra. Globális időpontok (STN modell). A téglalapok az általánosan hálózat topológiájú folyamat taszkjait jelölik.



a keresési tér. A diszkrét intervallumok esetével ellentétben nem lehetséges külső, időfüggő eseményekkel közvetlenül bővíteni a modellt.

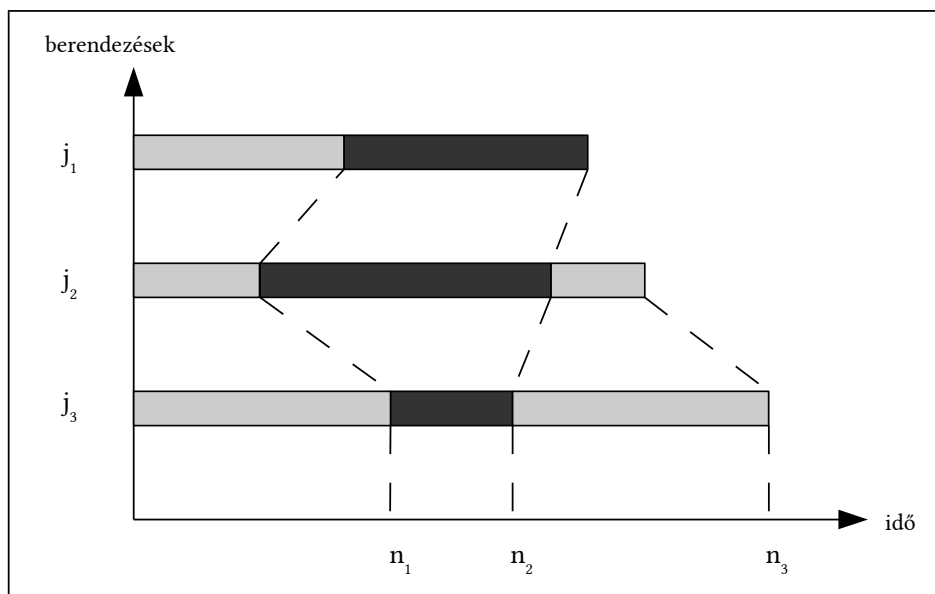
Berendezés-specifikus eseménypontos modellek. A berendezés-specifikus eseménypontos modellek a globális időpontok által okozott túlzott korlátozó hatást védik ki azzal, hogy berendezésenként külön időtengelyt vezetnek be, és ezeket szinkronizálják ([11],[12],[13]). Ezzel egy-egy intervallum berendezésenként máskor kezdődhet, illetve érhet véget, ez a rugalmasság a berendezés-specifikus modellek legnagyobb előnye. A döntési változók ugyanazok, mint diszkrét vagy globális időpontok esetén, azzal a különbséggel, hogy szétválík külön változókbá a taszkok berendezésekhez, illetve intervallumokhoz rendelése.

A modellek hátránya, hogy általában gyengébbek az LP-relaxációik, ezért alapvetően lassabbak. Ezt orvosolni lehet élesítő korlátozások beiktatásával ([12]).

Csoportosítás a berendezések reprezentációja szerint

STN modellek. Az STN és RTN modellek fogalma nem keverendő a feladattípusoknál említett reprezentációs lehetőségekkel. Feladatokat lehet

3.3. ábra. Berendezés-specifikus eseménypontok. Az egyes események berendezésenként eltérő időpontokban is történhetnek.



reprezentálni STN vagy RTN formátumban, de egy modellezési kérdés, hogy milyen formában történik a feladat feldolgozása. Eszerint léteznek STN és RTN modellek.

Az STN modellek esetében a berendezések a döntési változók dimenzióját képezik. Nem pusztán azok a döntések, hogy egy taszkot adott időpontban végrehajtsunk-e, hanem az is, hogy ez a különböző berendezésekben egyenként megtörténjen-e. Az STN modelleknél ezért külön vannak kezelve a gyártás során létrejövő anyagok (state-ek, állapotok), illetve a berendezések. A publikált időalapú modellek többsége STN modell ([4],[8],[12],[10]).

RTN modellek. Az RTN modellek lényege, hogy a berendezések nyersanyagokként vannak kezelve, melyekkel együttesen erőforrásoknak hívjuk őket. Ez azon egyszerű megfontolásból valósítható meg, hogy egy szakaszos üzemű taszk valójában a kezdetekor elfogyaszja, majd a végén újratermeli a berendezést. RTN modell esetén minden taszknál kötött, hogy melyik berendezésben hajtható végre. Ha egy taszkhhoz eredetileg több berendezés is lehetséges volna, akkor annyi különböző taszkot vezet be helyette a modell. A döntési változók azt jelölik, hogy egy adott intervallumban egy adott taszk fusson-e.

Kétféle lényegesen különböző RTN megvalósításról beszélhetünk. Az egyikben az azonos berendezésekből annyi különböző erőforrást származtat a modell, ahány darab van az adott berendezésből. A másik esetben egyetlen erőforrás van bevezetve egy berendezés típusra. Az első megközelítést a továbbiakban "lágym RTN"-nek nevezzük, mivel megőrzi a berendezések egységét, míg az utóbbit "kemény RTN"-nek. Az irodalomban lágym RTN modelleket publikáltak, mi munkánk során teszteltük egy lágym modellnek a kemény RTN változatát is. Létezik diszkrét intervallumos ([7]), globális időpontos ([9]) és berendezés-specifikus RTN modell is ([13]).

Az RTN általános előnye az STN-nel szemben az egyszerűbb, és gyakran kevesebb változót és korlátozást tartalmazó, és emiatt gyorsabb modell. Hátránya, hogy a berendezés-specifikus információk számon tartása nehezebb, bár lágym RTN esetén továbbra is lehetséges marad.

Csoportosítás az intervallumokhoz kötés módja szerint

Kutatásunk során két, egymástól lényegesen eltérő fajtáját sikerült az időalapú modelleknek elkülöníteni. A különbség a döntési változóknak implementációjában rejlik. A taszkok intervallumokhoz rendelése (allokációja) többféleképp megvalósítható. Mivel nem találtunk az irodalomban precedenst erre a különbségtételre, az általunk bevezetett neveket használjuk.

Start/stop allokációs modellek. Ha a modellekben a taszkok és intervallumok közti relációt olyan bináris változók jelölik, amelyek megadják, hogy egy adott időpontban egy taszk elkezdődik-e, illetve egy másik változóhalmaz jelöli, hogy egy adott időpontban befejeződtek-e, akkor a modellt start/stop allokációs modellnek nevezzük. Ekkor, ha N intervallum van, ez $2N$ bináris változót jelent. A publikált időalapú modellek közül több ezt a módszert alkalmazza ([8],[12],[10]).

A legnagyobb előny ebben az esetben, hogy egy taszk tetszőleges számú intervallumon át tarthat, és ehhez képest viszonylag kevés a bináris változó. Hátrány, hogy a modell méretben sokkal nagyobb, több folytonás változót igényel, és az LP-relaxációja is gyengébb.

Delta allokációs modellek. Több időalapú modell alkalmaz azonban egy általában nagy deltával (Δ) jelölt konstanst ([9],[13]). Ennek lényege, hogy egy-egy taszknak legfeljebb Δ intervallummal később be kell fejeződnie. A döntési változók azok, hogy egy n intervallumtól egy n' intervallumig tart-e egy taszk ($n \leq n' \leq n + \Delta$). Ez megközelítőleg N intervallum esetén $N \cdot \Delta$ bináris változót jelent.

Ennek a megközelítésnek az előnye, hogy a modell általában kevesebb bináris változót tartalmaz (kis Δ esetén), kevesebb a folytonos változó, erősebb a relaxáció. Hátrány, hogy a keresési tér kis Δ értékre nagyon szűk lehet, nagy értékre pedig a modell változóinak a száma lehet elfogadhatatlanul nagy. Ebben az esetben a keresési tér szűkebb, mint a start/stop allokációs modellek esetében.

3.1.3. Precedencián alapuló modellek

A precedencia alapú modellekben az időtengely csak impliciten jelenik meg, a fő döntések azok, hogy a végrehajtandó taszkokat milyen előidejűség szerint lehessen végrehajtani. A precedencia két taszk közötti reláció, melynek jelentése, hogy az egyik vége után kezdődik-e a másik. A receptek általában teljesítendő precedencia relációkat jelölnek ki (például egy termék egymást követő szintjeit egymás után kell végrehajtani), és a döntési változók is azt jelölik, hogy egy taszk megelőz-e egy másikat.

A precedencia alapú modellek hátránya, hogy nem képesek általános (Network) ütemezési feladatok megoldására, ez a legnagyobb hátrányuk. Egy feladat csak akkor megoldható precedencia alapú modellekkel, ha speciális feltételeket teljesít. Ezek például, hogy a taszkok számának fixnek kell lennie, nem megengedett egy-egy batch szétválasztása és több darabban végrehajtása. Az időalapú modellekhez képest sokkal bonyolultabb a tárolókapacitás, a megújuló erőforrások modellezése, ez általában több bináris változót

igényel. További nehézséget jelent a termelési költségek, illetve a profit vagy a termelés optimalizálása. A legyártandó termékek számának is előre rögzítenek kell lennie.

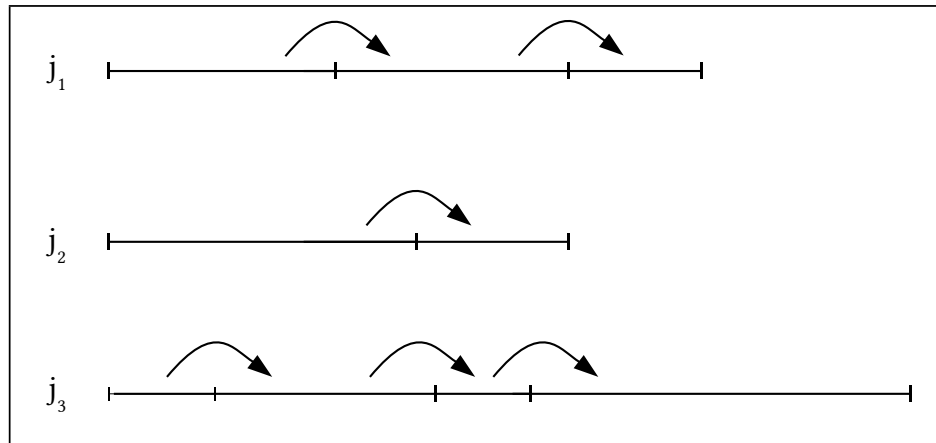
Előnye a precedencia-típusú megközelítésnek, hogy az időalapú modellekkel ellentétben nem szűkíti le a döntési teret, a modellek a multipurpose feladattípusra garantálják az optimalitást. Nincs szükség intervallumszámok vagy egyéb paraméterek megadására sem.

A precedencia alapú modellek elvben képesek Precedential típusú ütemezési feladatok megoldására. A gyakorlatban azonban ilyen feladat ritkán fordul elő, így a precedencia alapú modelleket egy speciálisabb, de gyakrabban előforduló feladatosztály, a multipurpose rendszerek megoldására írják. Háromféle precedencián alapuló megközelítés létezik:

- Berendezés-specifikus közvetlen precedencia
- Közvetlen precedencia
- Általános precedencia

Berendezés-specifikus közvetlen precedencia

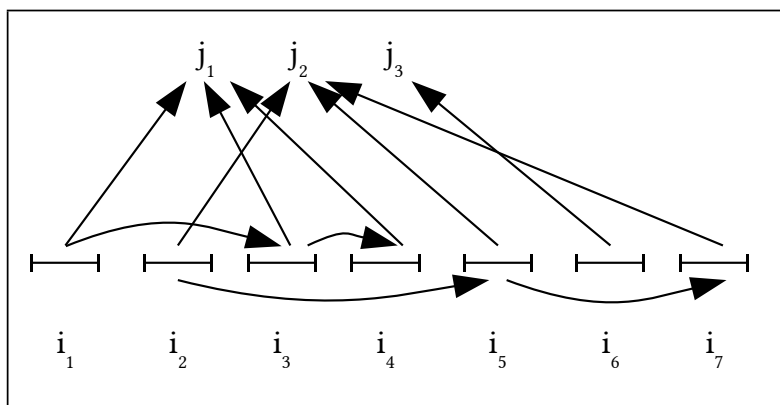
3.4. ábra. Berendezés-specifikus közvetlen precedencia. Minden berendezés taszkjai között előidejűségi sorrend áll fenn.



Megjelenését tekintve az első precedenciafogalom a berendezés-specifikus közvetlen precedencia volt ([14]). Ilyen modelleknél a döntési változóknak három dimenziója van: egy berendezésben egy adott taszkot közvetlenül kövessen-e egy másik adott taszk.

Közvetlen precedencia

3.5. ábra. Közvetlen precedencia. Szétválnak az előidejűség és a taszkok berendezésekhez rendelése.



A közvetlen precedencia fogalmánál ([15]) szétválnak a döntési változók. Külön bináris változó jelöli, hogy egy adott taszkhoz egy adott berendezést rendelünk-e, illetve hogy két adott taszk egymást közvetlen követi-e. Ezzel kisebb a modell mérete, mint a berendezés-specifikus modellek esetén.

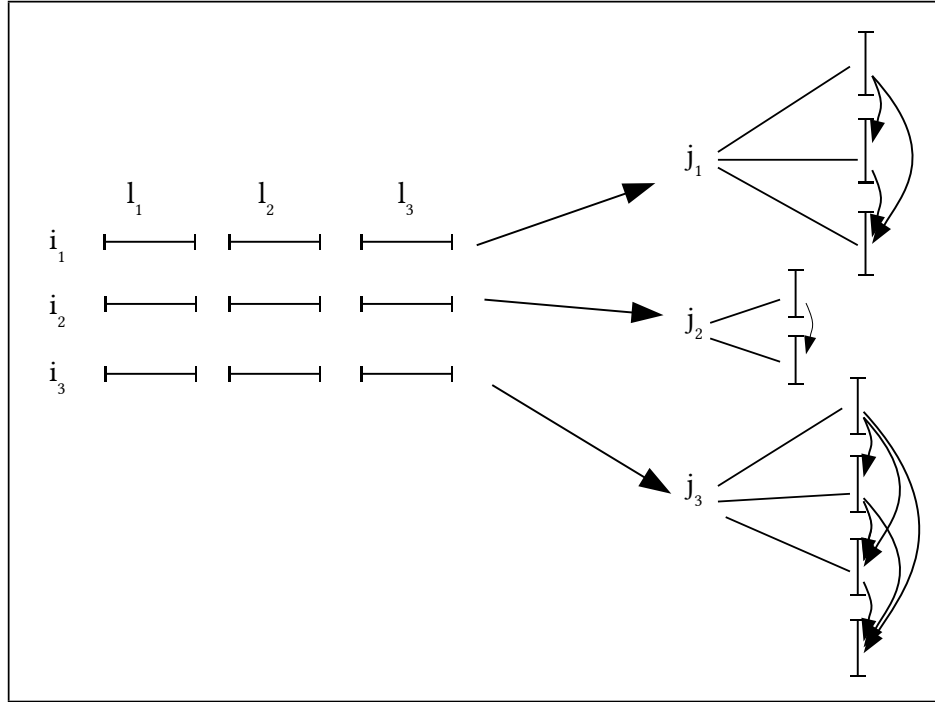
Általános precedencia

A legfrissebb és egyben legjobbnak bizonyuló precedencia-alapú modell az általános precedencia fogalmát használja ([16],[17]). Itt nem azt jelöli változó, hogy egy taszk közvetlenül megelőzi-e egy másikat, hanem hogy általában, megelőzi-e. Ezzel a módszerrel nincs több bináris változó, a modell mégis egyszerűbb és rugalmasabb. A precedencia modellek nagy hátrányát jelentő tárolást is némileg orvosolni tudja az általános precedencia használata azzal, hogy tároló taszkokat vezet be a modell.

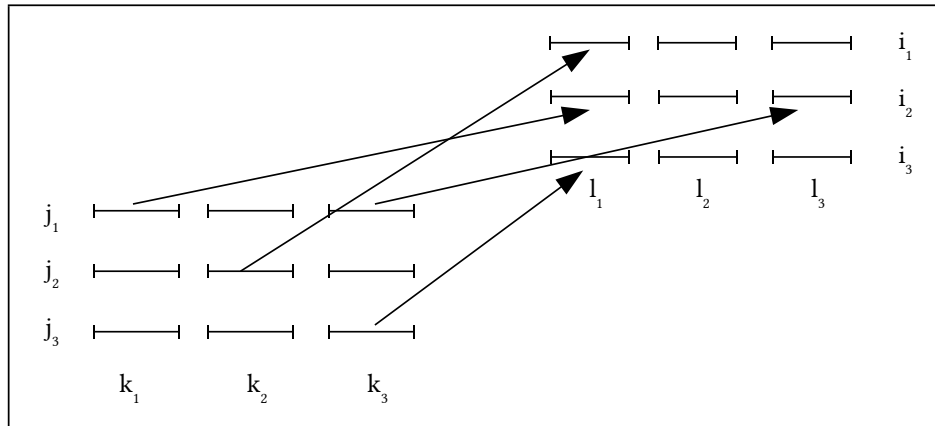
3.1.4. Slot alapú modellek

Az slot-ok vagy időszeletek alkalmazása a precedencián alapuló modellekkel rokon, ám a döntési változók területén lényeges a különbség. Egy slot-alapú modell ([18],[19]) speciális multipurpose modelleket tud megoldani, ahol a berendezéseket tekintve nincs választási lehetőség (tehát minden termék minden szintjén egy berendezés áll). Ezen feltételek mellett fix egy berendezés

3.6. ábra. Általános precedencia. Minden közvetett, vagy közvetlen előidejűséget bináris változó jelez.



3.7. ábra. Slot alapú modellek. A berendezések időszeteinek és a multipurpose folyamat taszkjainak a megfeleltetése történik.



által végrehajtandó taszkok halmaza. A legyártandó termékeknek és a berendezéseknek is megfeleltet rögzített számú időszeletet, és a modell ezeket felelteti meg egymásnak. A modellezési változók jelzik, hogy egy termékhez tartozó időszelet és egy berendezéshez tartozó időszelet összetartozik-e.

Az slot alapú modelleknek elméletileg csak hátránya van a precedencián alapuló modellekkel szemben. Lényegesen szűkebb feladatostályt tud megoldani, a bináris változók száma sokkal nagyobb, és a modell relaxációja is gyengébb.

3.1.5. Egyéb megközelítések

Léteznek az előbb felsoroltakon kívüli matematikai modellek is. Ezek általában nagyban hasonlítanak valamely említett módszerre, de logikájukban mégis különbözhetnek. Ezek közül az SSN reprezentációt mutatjuk be. Az állapotsorozat-hálózat (State-Sequence Network, SSN [20]) lényege, hogy magának a feladatnak a reprezentációjának a megfogalmazása is kizárólag állapotokkal történik. Ebben az esetben a berendezések, taszkok csak implicit módon, halmazokkal szerepelnek a modellben, minden döntési változó az ütemezett folyamat alapanyagaira és termékeire, vagyis állapokra vonatkozik. Az idő reprezentációja berendezés-specifikus eseménypontokkal történik. Nem egy berendezést használunk egy eseménypontban, sem egy taszkot hajtunk végre, hanem egy állapottal történik esemény, ezek a döntési változók.

A bináris változók száma valamivel kevesebb, mint a hasonlóan működő globális idő- vagy eseménypontokat alkalmazó modellek esetén, ez a módszer előnyének tulajdonítható. Hátrányt jelenthet a bonyolultabb reprezentáció. Megjegyzendő, hogy tetszőleges topológiájú ütemezési feladatot le lehet írni SSN-moddal.

Az SSN-módszer létezése azt sugallja, hogy a szakaszos ütemezési feladatok MILP modellekkel való megoldásában lehetségesek újabb típusú modellek, ám ezek a korábbi lényeges ötletek átdolgozásából, továbbfejlesztéséből jöhetnek csak létre.

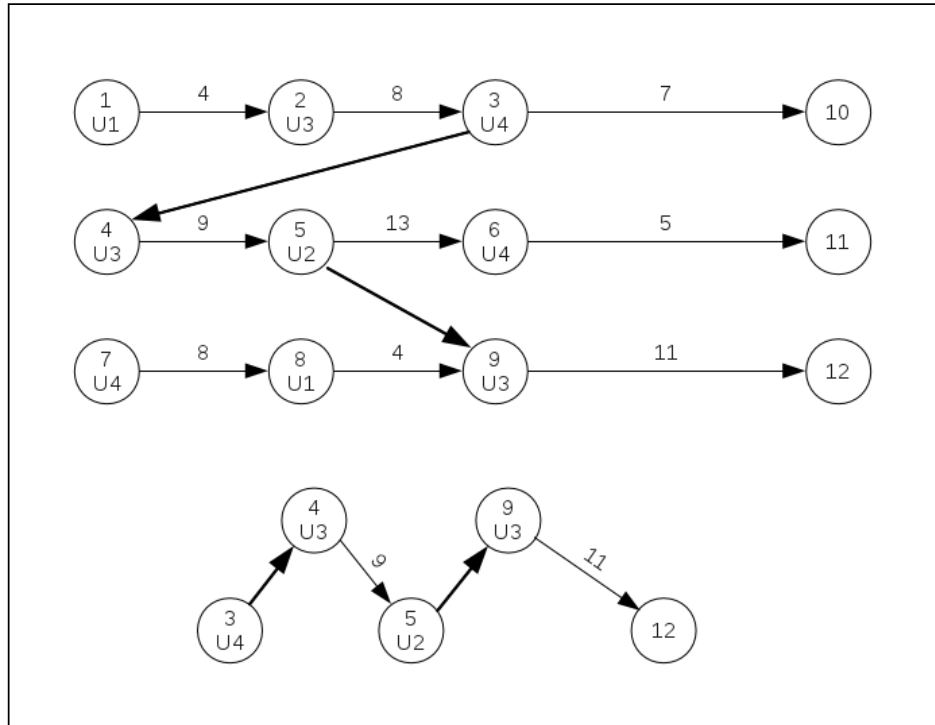
3.2. Kombinatorikai megközelítések

3.2.1. S-gráf módszertan

Tegyük fel, hogy az ütemezendő taszkok halmaza rögzített. Feleljen meg minden taszknak és készterméknek egy-egy csúcs. Minden, taszkot reprezentáló csúcsból induljon irányított él azon taszkok által jelzett csúcsokba, amelyek a

gyártási folyamat során közvetlenül követik (vagyis közvetlenül felhasználják a taszk által készített köztes terméket), és azon késztermékek által jelzett csúcsokba, amiket a taszk termel.

3.8. ábra. Példa S-gráfra. A termékek és a berendezések taszk-sorrendjét is súlyozott, irányított élek jelzik.



Minden irányított élhez egy nemnegatív súlyt rendelünk, melynek értéke az él kezdőpontjához tartozó taszk végrehajtási ideje. Ezt a reprezentációt nevezzük ütemezési gráfnak, vagy S-gráfnak (Schedule Graph) [21]. A fogalmakhoz matematikailag igazolt működésű algoritmusok tartoznak, melyek együttesen alkotják az S-gráf módszertant [2]. A megvalósítható ütemezéseket jelölő S-gráfok halmaza ugyanis bejárható, például a B&B keretalgorithmus által. Kombinatorikai algoritmusokkal kezelhető a korlátozás (Bounding) eljárás, és az ütemezés megvalósíthatóságának ellenőrzése. Számos gyorsító eljárás létezik [22], melyekkel a módszertan kiterjeszthető ipari méretű feladatok megoldására is.

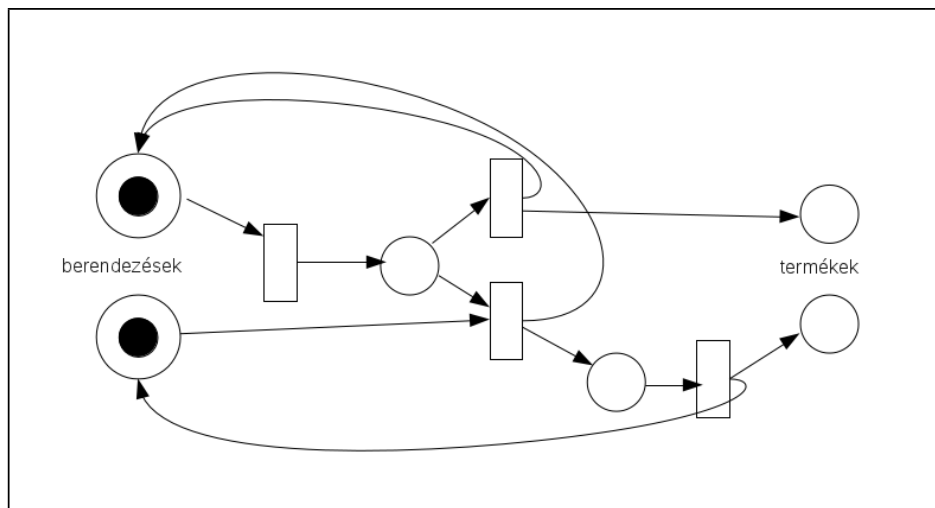
Az S-gráf módszertan alkalmas minden precedential típusú feladat megoldására, ebben megegyezik a precedencián alapuló moellek

képességeivel. Gyengesége, ha a feladatban szerepelnek speciális korlátozások (például megújuló erőforrások, véges köztes tárolókapacitás, lehetőség gyártási szakaszok szétválasztására és keverése egy taszkon belül). Az S-gráf kedvező tulajdonsága, hogy könnyen kizárhatóak az NIS tárolási stratégiánál adódó „cross-transfer”-ek, melyek a matematikai programozási modelleknél komony problémát jelentenek. Munkánk során multipurpose ütemezési feladatokra teszteltük a módszer hatásosságát.

3.2.2. Időzített állapotú Petri hálók

Egy lehetséges megközelítés a szakaszos ütemezési feladatok megoldásához a Petri hálók fogalmának kiterjesztése. Az idő, mint tényező az ütemezés során megkerülhetetlen, ezzel a céllal feltétlenül be kell építeni az alapvetően analitikus célokat szolgáló Petri hálókba. Erre két lehetséges megoldás van: vagy a tranzíciókhoz rendelünk késleltetéseket (Timed Transition Petri net, TTPN), vagy az állapotokhoz (Timed Place Petri-net, TPPN). Az előbbi esetben a tranzíció tüzelése és a kimenet megjelenése között, míg az utóbbinál a megjelenő kimeneti token(ek) és azok újbóli felhasználása között vannak kikényszerített késleltetések.

3.9. ábra. Példa Petri hálóra. A tranzíciók által kell elérni, hogy a termékek végállapotaiba tokenek kerüljenek.



sok (nyersanyagok, termékek, berendezések) állapotokkal, a taszkok tranzíciókkal vannak reprezentálva a Petri hálóban. Egy erőforrás elérhetőségét az állapotában lévő token jelzi. Az Időzített állapotú Petri háló állapotaihoz a tokenek helyzete mellett az elérési idők is hozzátartoznak. B&B keretprogrammal megtalálható az optimális megoldás a Petri háló „reachability”-fájában, amely tartalmazza az ütemezési döntésekkel elérhető állapotokat.

A megközelítés legnagyobb előnye, hogy képes kezelni diszkrét erőforrásokat, berendezéseket, lényegében RTN reprezentációval analóg módon. Petri háló használata során a „cross-transfer”-ek is kiszűrhetők. Bár a módszer ígéretes, nagyméretű, ipari ütemezési feladatban való alkalmazására eddig még nem volt példa.

3.2.3. Időzített automaták

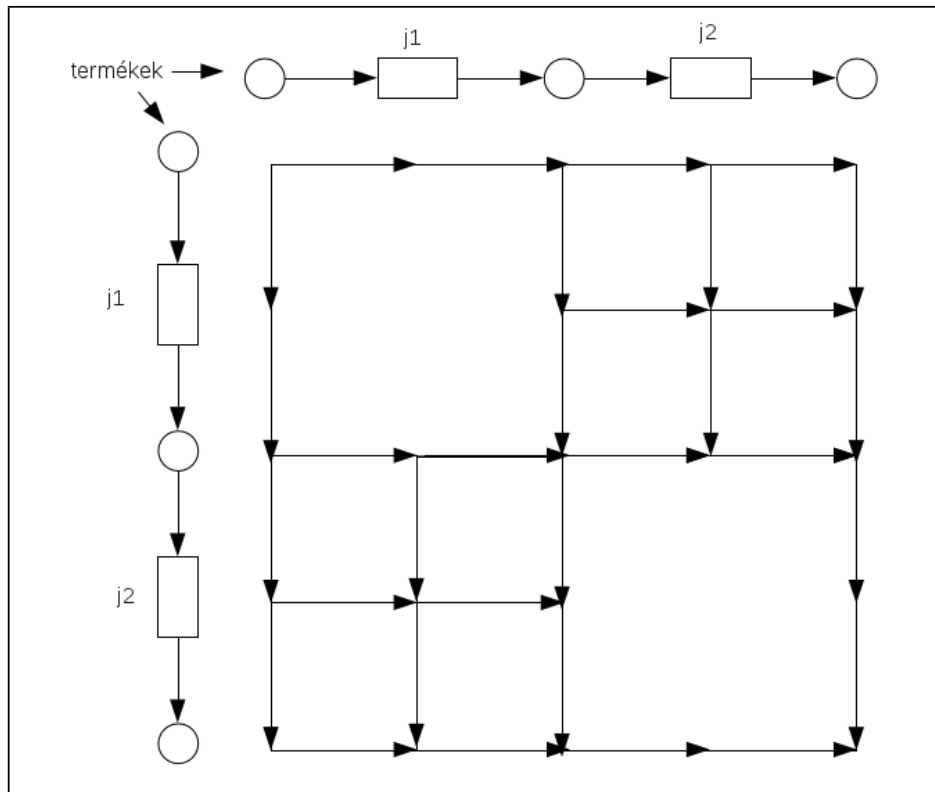
Tegyük fel, hogy egy véges automatához folyamatosan járó órák tartoznak, melyekkel két műveletet lehet végezni: leolvasni az értéküket, vagy alap helyzetbe állítani (nullázni) a számlálójukat. Az automatán minden műveletet csak az órák aktuális értékeire felírt lineáris egyenlőtlenségek, más néven időkorlátozások teljesülése esetén lehet elvégezni, valamint az automata állapotaira is időkorlátozásoknak kell teljesülni. Minden művelettel együttjárnak bizonyos akciók, amelyek órák adott halmazának lenullázásai. Az állapotokhoz az órák ideit mutató vektor is hozzátartozik. A véges automaták így módon kiterjesztett változatát időzített automatáknak nevezzük (Timed Automata, TA) [24].

Időzített automatákkal egy többcélú ütemezett folyamat állapota leírható [25]. Minden terméknek és erőforrásnak egy önálló automata felel meg, melynek állapotai az adott termék gyártásának előrehaladottságát, vagy az erőforrás elérhetőségét jelképezi. Órák tartoznak minden termékhez, amellyel a taszkok végrehajtását időben korlátozni lehet. Az önálló automaták műveletei közősek is lehetnek, amennyiben egy adott erőforrás (például berendezés) felhasználását egy termék taszkja igényli. Az önálló automaták párhuzamos kompozíciója az összes órával együtt egy időzített automatát eredményez, melynek állapotai jelzik az ütemezett folyamat állapotait.

Ezen automaták használhatóak ütemezési feladatok megoldására, optimalizálásra. Ez megtörténhet a kezdőállapotból indulva, az automata állapotai állapotgráfjának műveleteken keresztüli bejárására, megkeresve az optimális megoldást. Lehetőség van a keresési teret csökkenteni, például a B&B algoritmusmal analóg módon korlátozó függvényt definiálni egy adott állapotból elérhető legjobb ütemezés megbecslésére. Ez az árazott időzített automaták fogalmához vezet.

A módszer előnye, hogy képes RTN reprezentációval modellezni mul-

3.10. ábra. Példa időzített automatára. Az egyes berendezések (és az ábrán be nem rajzolt erőforrások) határozzák meg az automata állapotainak „dimenzióit”.



tipurpose feladatokat, ezen kívül sok valós korlátozás beépíthető a modellbe. Hatékonyabb az algoritmus, ha minden taszkhoz csak egy végrehajtó berendezés tartozik. Hátránya, hogy a bejárasi algoritmus során, a keresési teret drasztikusan csökkentő eljárások nélkül, a vizsgált állapotok száma beláthatatlanul nagy lehet. Ezen kívül egy-egy termékfolyamatnak a felosztása és újraegyesítése nem kezelhető időzített automatákkal, tehát a taszkok bemenetének mérete rögzített kell, hogy legyen.

4. fejezet

Egységes formátum ütemezéshez

Munkánk eredménye egy, az ütemezési feladatokhoz tartozó általános leíró formátum, és az erre épülő, GNU MathProg kódrészletekből álló rendszer (úgynevezett wrapper), amely:

- Áthidalja a különböző ütemezési feladattípusok közti adatkonverzió problémáját;
- Lehetővé teszi az ütemezési feladattípusok és a különböző fajtájú MILP modellek fejlesztésének elkülönítését;
- Automatizálja az egyes modellek tesztelését a különböző feladatokon.

A most következő fejezetben az elkészült wrapper funkcióját ismertetjük.

4.1. Felhasznált nyelv

Munkánk során a GNU MathProg leírónyelven implementáltuk a modelleket, és a tesztelt feladatokat. Egy segédprogram, a *glpsol* segítségével egy-egy, az adott nyelven írt modell- és adatfájl együtteséből az általános célú MILP szolverek bemenetül szolgáló struktúrát generál. A segédprogram a modellfájlban leírt paraméterek értékeit az adatfájlból nyeri ki, így különül el a modell, és a konkrét feladat. Az ütemezési MILP modellt a modellfájlban, az ütemezési feladatot az adatfájlban kell implementálni.

Az egységes formátum azt hivatott megkönnyíteni, hogy a modellek implementációjakor ne kelljen figyelembe venni, hogy éppen milyen feladatra akarjuk tesztelni a modellt. Ugyanez a cél fordítva is: egy-egy ütemezési feladatot egyszer kelljen csak leírni, akkor is, ha később több különböző modellel akarjuk megoldani.

4.2. Feladatok egységes formátuma

Az öt feladattípus (Single stage, Multiproduct, Multipurpose, Precedential, Network) leírása eltérő paraméterek megadását követeli meg, amely öt lényegesen különböző feladatléírási módokhoz vezet. Mivel ezek egymásba ágyazódnak, egy Network problémákat megoldó modell elvben gond nélkül meg tudna oldani egy speciálisabb, például Multipurpose feladatot is. Ennek akadálya az, hogy egy Network típusú modell implementációja olyan bemeneti adatot vár, amely egy általános, hálózat-topológiájú reprezentáció. A speciálisabb feladat, hogy megoldható legyen át kell írni az általánosabb formátumba.

Első lépésként kijelöltük és rögzítettük az öt feladattípus bemeneti formátumát. Konvencióként megfogalmaztuk, hogy minden feladatot annak a legspeciálisabb feladattípusnak a formátumában rögzítünk, amelybe tartozik. Ezután implementáltuk a wrappert, amely elvégzi a különböző feladattípusokban rögzített adatok beolvasását. Speciális feladat leíró formátumából képes az általánosabb típus adatszerkezeteibe konvertálni. Fordítva nem, de erre nincs is szükség, mert egy modell nem is képes egy, általánosabb feladattípus megoldására.

4.3. Modellek implementációja

Az implementálandó MILP modellek a wrapper által deklarált paraméterekkel dolgoznak.

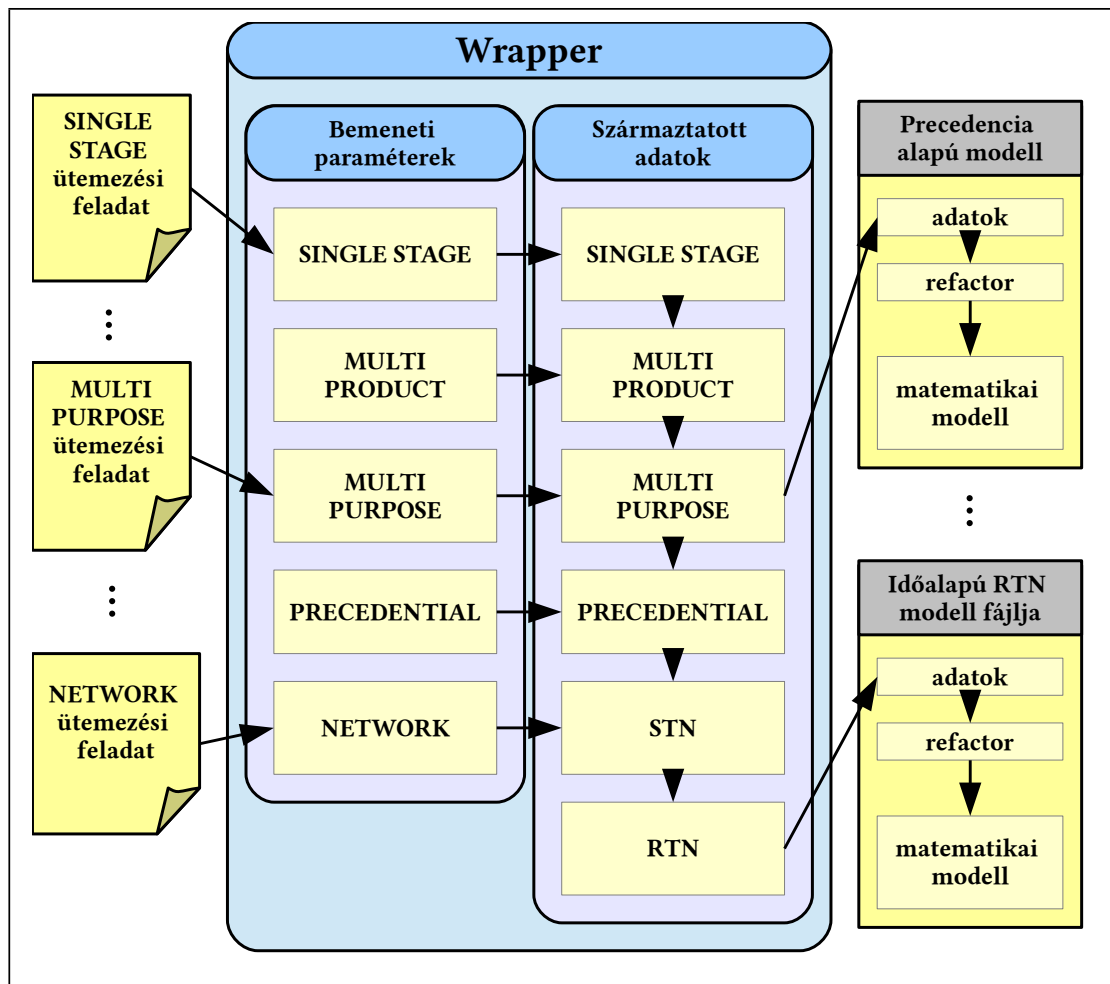
A wrapper technikailag egy fejrész, melyet egy script segítségével a modellfájl elejére kell illeszteni a segédprogram futtatása előtt. A modellfájlban a wrappert követi egy átnevező rész (refactor), amely átnevezi és végleges, olyan reprezentációs formára hozza a feladatot, amelyet a matematikai modell megkíván. Ezek után egy MILP modell implementációja már tetszőleges változónevekkel megvalósítható, így gond nélkül használható azon irodalmak nevezéke, amelyekből a modell származik.

Fontos megjegyezni, hogy a wrapper által generált reprezentáció-formák halmaza nem egyezik meg a feladattípusok halmazával. Egy Network típusú feladatot, mely például az egységes formátum alapján STN reprezentációval van megadva, a wrapper kifejt RTN megvalósításban is. Ezen kívül, a wrapper több, a beolvasásnál még nem megjelenő adatszerkezetet generál, amelyekre a különböző modelleknek szüksége lehet.

A wrapperrel ellátott modellek letölthetők a későbbiekben bemutatott, nyilvános feladatadatbázisunkból. Segítségével minden implementált modell további változtatások nélkül tesztelhető bármely feladaton, amely nem ál-

talánosabb formátumú, mint amelyet a modell elméletileg meg tud oldani.

4.1. ábra. Wrapper - az egységes formátumot kezelő rendszer. Segítségével egy feladat adatfájlja tetszőleges, a megoldására képes módszer modellfájljával együtt fordítható egy segédprogrammal, automatizálva ezzel a tesztelést. A nyilak jelölik az adatok lehetséges áramlási útját



A wrapper forrása letölthető a 4.2. ábrán látható QR kód segítségével, amely a következő URL-re mutat: <http://scheduling.dcs.uni-pannon.hu/wrapper/>.

4.2. ábra. QR kód a wrapper forrásához



5. fejezet

Korábbi tesztek

Korábbi munkánk során már összehasonlítottuk az általános célú MILP szolvereket:

- GLPK
- CBC
- Gurobi
- CPLEX

A tesztek a következő felszereltségű laptopon végeztük:

- Intel Core i5-520M CPU (2 mag, 2,4 GHz)
- 4 GB DDR3 memória (1066 MHz)

Feltérképeztünk alapvető különbségeket az egyes MILP modelltípusok és az S-gráf módszertan között. Korábbi munkánk részeként az alábbi modell-összehasonlításokat végeztük el:

- Globális időintervallumos RTN és STN modellek futási eredményeit vetettük össze. Tesztjeinkből megállapítottuk, hogy az RTN modellek általában gyorsabbak, de néhány szélsőséges esetre nem működnek.
- Globális időintervallumos és berendezés-specifikus modellek futási eredményeit hasonlítottuk össze. Ebből kiderült, hogy a berendezés-specifikus eseményekkel valóban jobban kihasználható a keresési tér, de változó, hogy mely modellek és melyik szolverek párosítása a leggyorsabb

- Precedencián alapuló és időalapú modellek futási eredményeit hasonlítottuk össze. Igazoltuk, hogy a precedencián alapuló modellek garantálják az optimális makespan értéket. Erősen testesetfüggő, hogy melyik megközelítés eredményez gyorsabb megoldást, az egyenlően elosztott batchszámoknál a precedencián alapuló modellek, míg a szélsőségesen egyoldalú teszteseteknél az időalapú modellek bizonyultak eredményesebbnek.
- Az S-gráf módszertan és az ahhoz tartozó, tanszéki fejlesztésű megoldó szoftver teljesítményét összehasonlítottuk az irodalomban legnépszerűbb MILP modellek eredményeivel. Ebből kiviláglott, hogy az S-gráf módszertan multipurpose feladatokra gyors és optimalitást tud garantálni, nagyobb feladatoknál ez az előnye elvész, és a részeredményekben is gyengébb lesz a MILP modelleknél.

Tapasztaltuk, hogy időalapú modellek esetén létezik egy minimális intervallumszám, amelyre a feladat megoldható, de az alatt még az LP relaxációk sem adnak megoldást. A kis intervallumszám valóban leszűkíti a keresési teret, ez meglátszott a modellek eredményeinek szuboptimalitásában is. Az intervallumszám növelésével a talált célfüggvény-értékek mérsékelten javultak, viszont a szükséges futásidő drasztikusan romlott.

Futási eredményeinkből kiderült, hogy a tesztesetek méretének növelésével (vagyis a batchszámok emelésével) a problémák megoldási ideje fokozatosan romlott, bár voltak kivételek, amikor nagyobbnak ítélt feladatot ugyanaz a megoldó, ugyanazzal a modellel gyorsabban oldott meg. Megállapítottuk, hogy nagy feladatoknál az 1000 másodperc alatt talált legjobb célfüggvényértékek (makespan) a kis eseteknél gyorsabb szolver esetében jobbak voltak.

Mindezek alapján új tesztelési módszert alkalmaztunk. Jelen vizsgálatainkban a figyelmet a kisebb tesztesetekre összpontosítottuk. Teszteltünk a korábban jónak bizonyult modellek mellett három újat ([19],[10],[12]).

6. fejezet

Empirikus vizsgálatok

6.1. Modell adatbázis

Számos modellt gyűjtöttünk össze különböző tudományos folyóiratokból, majd a 3. fejezetben bemutatott kategóriák szerint csoportosítva, adatbázisba szerveztük ezeket. Az adatbázist egy webalkalmazás segítségével kezelhetjük, mely a korábban általunk készített feladat adatbázist bővíti ki. Az alkalmazás a Django Python-alapú keretrendszerre épül, az adatokat a MySQL relációs adatbáziskezelő szoftver segítségével tárolja.

Az alkalmazás lehetőséget biztosít a feldolgozott modellek listázására, megtekintésére, illetve szűrésére a 3. fejezet kategóriái szerint. Az egyes modellekhez csatolható egy rövid, szöveges leírás is. Az általunk implementált modellek le is tölthetőek az egységes formátumban.

Az adatok feltöltését egy intuitív adminisztrátori felület teszi lehetővé. Ennek köszönhetően egyszerűen adhatunk hozzá adatbázisunkhoz új modelleket.

Az alkalmazás a <http://scheduling.dcs.uni-pannon.hu/> URL-en érhető el.

6.2. Tesztkörülmények, módszerek

A tesztek a következő felszereltségű laptopon végeztük:

- Intel Core i5-520M CPU (2 mag, 2,4 GHz)
- 4 GB DDR3 memória (1066 MHz)

Az operációs rendszer mindegyik fizikai magot két logika CPU-nak tekinti, tehát alkalmas arra, hogy a párhuzamosított megoldók fölénye megmutatkozzon.

6.1. ábra. Az adatbázisban lévő modellek listázása

scheduling of batch processes			
HOME PROBLEMS MODELS ABOUT LINKS			
Published in	Capability	Type	Download
Carlos Alberto Méndez and Jaime Cerdá 2003 - An MILP Continuous-Time Framework for Short-Term Scheduling of Multipurpose Batch Processes Under Different Operation Strategies	Multipurpose	Precedence based	↓
Sang Beom Kim and Ho-Kyung Lee and In-Beum Lee and Euy Soo Lee and Bomsock Lee 2000 - Scheduling of non-sequential multipurpose batch processes under finite intermediate storage policy	Multipurpose	Slot based	↓
Sundaramoorthy, A. and Karimi, I. A. 2005 - A simpler better slot-based continuous-time formulation for short-term scheduling in multipurpose batch plants	Network	Time based	↓
Castro, P. M.; and Barbosa-Póvoa, A. P.; Matos, H. A. and Novais, A. Q. 2004 - Simple Continuous-Time Formulation for Short-Term Scheduling of Batch and Continuous Processes	Network	Time based	↓

Scheduling

6.2. ábra. A webalkalmazás elérhetősége



A használt operációs rendszer az Ubuntu 12.04 Linux disztribúció 32-bites verziója volt.

Az S-gráfot alkalmazó modelleket egy több szálon dolgozó tanszéki megoldóval oldottuk meg [26].

A különböző lineáris programozási módszereket alkalmazó modelleket a következő megoldókkal teszteltük:

- Gurobi 4.5.1
- CBC 2.5.0

Ahol ezt másként nem jelezzük, ott a szoftver alapbeállítását használtuk.

A használt szoftverek egy MILP modell megoldását csupán addig javítják, amíg az optimális megoldás felső- és alsó korlátjának hányadosa nagyobb, mint 1,0001.

Amennyiben a megoldó a rendelkezésére álló idő alatt nem találta meg az optimális megoldást, akkor az addigi legjobbat tekintettük eredménynek. Ha nem talált lehetséges megoldást, akkor nem adtunk meg eredményt.

A CBC¹ a Computational Infrastructure for Operations Research (COIN-OR) projekt részeként fejlesztett nyílt forráskódú MILP megoldó. Binárisként az Ubuntu által szállítottat használtuk.

A Gurobi Optimization Inc. által fejlesztett Gurobi² 4.5.1-es verzióját használtuk. A gyártó ajánlásának megfelelően³ nem alkalmaztunk a gyáritól eltérő beállítást.

Előző munkánkban [27] bemutattuk, hogy a nyílt forrású GLPK⁴ teljesítménye messze elmarad a szintén szabad szoftver CBC eredményeitől, ezért annak ellenére, hogy elterjedt megoldó, nem szerepel az összehasonlításban.

Az IBM ILOG CPLEX Optimizer⁵ még a GLPK-nél is népszerűbb, azonban teljesítménye nagyságrendileg megegyezik a Gurobiéval. Bizonyos modellek esetén gyorsabbnak bizonyult a Gurobinál, más esetekben az állítás fordítottja igaz. Emiatt úgy döntöttünk, hogy szerepeltetése nem adna további mélységet analízisünknek – ezért nem szerepel összehasonlításunkban.

Minden modellt háromszor oldottunk meg a megoldókkal, majd a három futási idő számtani közepét vettük.

¹A CBC hivatalos weboldal: <https://projects.coin-or.org/Cbc>

²A Gurobi hivatalos weboldala: <http://gurobi.com>

³Video on Improving the Performance of the Gurobi Optimizer – <http://www.gurobi.com/resources/seminars-and-videos/improving-the-performance-of-the-gurobi-optimizer>

⁴A GLPK hivatalos oldala: <https://www.gnu.org/s/glpk>

⁵A CPLEX hivatalos weboldala: <http://www.cplex.com>

6.2.1. Esettanulmányok

Négy feladatot vizsgáltunk számítógépes tesztjeink során:

- Egy kisméretű, saját készítésű multiproduct feladat az egyes modellek keresési terének feltérképezésére;
- Egy közepes méretű multiproduct feladat, a modellek általános összehasonlítására;
- Egy közepes méretű multipurpose feladat, mely méretét tekintve a másodikkal rokon;
- Egy nagyméretű ipari feladatot, sok berendezéssel, bizonyos modellpárok összehasonlítása végett.

Az alábbi modelleket teszteltük:

- Egy precedencián alapuló modell multipurpose feladatok megoldására ([17]), a továbbiakban **Mendez2003**. A modellt egy olyan korlátozással bővítettük, amellyel a modell még jobban kihasználja a feladattípus adottságait, ezáltal sokkal gyorsabb.
- Egy slot alapú modell, speciális multipurpose feladatokhoz ([19]), a továbbiakban **Kim2000**. A modellhez a relaxációt javító, gyorsító hatású korlátozásokat adtunk.
- Egy időalapú, start/stop allokációs, globális időpontos STN modell ([10]), a továbbiakban **Sundaramoorthy2005**.
- Egy időalapú, start/stop allokációs, berendezés-specifikus eseménypontos STN modell ([12]), a továbbiakban **Janak2004**.
- Egy időalapú, delta allokációs, berendezés-specifikus eseménypontos RTN modell ([13]), a továbbiakban **Shaik2009**.
- Egy időalapú, delta allokációs, globális időpontos RTN modell ([9]), melyet implementáltunk a „lágy RTN” és a „kemény RTN” fogalmának megfelelően is, a továbbiakban **Castro2004L**, **Castro2004K**.
- Tanszéki fejlesztésű S-gráf megoldó

Teszteltünk UIS és NIS tárolási stratégiák mellett is.

Célunk (az első esettanulmány kivételével) az egyes modellek futási idejének összehasonlítása volt.

6.1. táblázat. Adatok az I. esettanulmányhoz - berendezések és végrehajtási idők.

Termék	1. lépés	2. lépés	3. lépés
A	(E1, 1)	(E2, 6) (E3, 4)	(E4, 1)

6.3. Esettanulmány I.

A felhasznált kisméretű, multiproduct feladat. Egy termék, A elkészítéséhez rendre három szint szükséges, összesen négy berendezéssel. A második szinten választási lehetőség van. A feladat minden esetben a makespan minimalizálása volt, két batch A termék előállítása mellett.

Erre a feladatra az összes vizsgált modellt alkalmaztuk, mindkét szolver lényegében egy pillanat alatt lefutott egy ilyen kis méretű feladatra, NIS tárolási stratégia mellett.

Futási eredmények. A lényegi eredmények a talált célfüggvényértékekben (makespan) nyilvánulnak meg, ugyanis kiderül, hogy a modellek milyen mértékben szűkítik le a keresési teret. a 6.5 táblázat összefoglalja a futási eredményeket.

Értékelés. Az optimális célfüggvényérték: $MS=8$, melyre egyetlen megoldás létezik. Látható, hogy a precedencia alapú Mendez2003 modell gond nélkül megtalálta az optimumot.

Időalapú modelleknél az intervallumszám előre megadandó, és ez nagyban befolyásolja a modell keresési terét. A start/stop allokációs időalapú modellek (Sundaramoorthy2005, Janak2004) egyformán teljesítettek, attól függetlenül, hogy globális, vagy berendezés-specifikus eseményeket használnak. Négy intervallumra még a gyenge $MS=10$ eredményt találják meg, de öt intervallum felett már az optimális megoldást.

Az időalapú modellek közül a delta allokációs modellek (Castro2004, Shaik2009) teljesítménye erősen függ az intervallumszámtól és Δ értékétől. Megfigyelhető a különbség a globális időpontokat használó Castro2004 és a rugalmasabb, berendezés-specifikus Shaik2009 között. A Castro2004 azonos intervallumszám és delta mellett is gyengébb eredményt ad, mint a Shaik2009, de az optimális megoldást ez utóbbi sem találja meg $\Delta < 1$ esetén. Mindkét modell az optimumát legalább $\Delta \geq 2$ esetén és minimum 5 intervallumnál éri el. Ebből látszik, hogy az intervallumszám és esetlegesen Δ növelése a megoldás optimalizálása szempontjából gyakran szükséges.

6.2. táblázat. Adatok a II. esettanulmányhoz - berendezések és végrehajtási idők.

Termék	1. lépés	2. lépés	3. lépés
A	(E1, 8)	(E4, 5)	(E5, 3)
B	(E1, 7)	(E3, 3)	(E5, 4)
C	(E2, 6)	(E4, 9)	(E5, 3)
D	(E2, 4)	(E3, 6)	(E5, 4)

6.4. Esettanulmány II.

A vizsgálat alanya egy irodalmi példa [28], mely közepes méretű multiproduct feladatnak számít. Négy termék elkészítése történik egy 5 berendezésből álló üzemben, melyek sorrendje kötött, azonban egy-egy terméknek csak 3-3 meghatározott berendezésen kell átmennie. A taszkok előre rögzítettek. A berendezések kötött sorrendjéből adódóan minden végrehajtandó taszkhoz csak egy lehetséges berendezés tartozik. A példa receptjét a 6.3 táblázat tartalmazza.

A feladat rendre a Castro2004L, Mendez2003, Sundaramoorthy2005, Shaik2009, és az S-gráf modellekre lettek tesztelve. A batchméretet folyamatosan növeltük, amíg a modell nem adott kétszer egymás után időlimit túllépést. A következő táblázatok foglalják össze a futási eredményeket: 6.6 - 6.7. A tesztesetek nevében a megfelelő modelleknél szerepel az intervallumszám (TP) és Δ értéke.

Értékelés. Megfigyelhető, hogy a precedencián alapuló MILP és az S-gráf szolver időben és optimalitásban felülmúlja az időalapú modelleket. Ennek oka valószínűleg az, hogy a feladat speciális, és ezek a modellek jobban kihasználják a struktúráját. A saját gyorsításunkkal ellátott precedenciális modell, a Mendez2003 az optimális megoldást adta, és a legnagyobb tesztesetekre futottak le a modellek közül.

Érdekes megfigyelni az időalapú modellek eredményeit is. Legjobban a Castro2004L teljesített, de teljesítménye nagyon egyenetlen, a start/stop típusú Sundaramoorthy2005 és a delta allokációs, de berendezés-specifikus Shaik2009 modellek teljesítménye viszont a batch számok növelésével egyenesen romlott.

Mindkét modell, a Sundaramoorthy2005 és a Castro2004L is ugyanazt a legnagyobb tesztesetet (5-5-5-5) voltak képesek megoldani. A berendezés-specifikus Shaik2009, bár azonos intervallumszám mellett gyakran jobb makespan eredményt talált, futási időben gyengébbnek bizonyult a globális időpontos modelleknél.

6.3. táblázat. Adatok a III. esettanulmányhoz - berendezések és végrehajtási idők.

Termék	1. lépés	2. lépés	3. lépés
A	(E1, 6)	(E3, 9)	(E4, 7)
B	(E2, 9)	(E3,15)	(E4,17)
C	(E1, 8)	(E4,14)	(E2,16)
D	(E2, 7)	(E3,11)	(E1, 4)

A Kim2000, az egyetlen slot alapú modell nagyon gyenge eredményeket ért el, összesen két tesztesetre futott le. Ezzel ki lehet jelenteni, hogy a slot alapú modellek jelen formájukban nem vehetik fel a versenyt a többi módszerrel.

A Janak2004 szintén nagyon gyenge teljesítményt nyújtott, egyetlen tesztesetre futott le. Ez azt sugallja, hogy berendezés-specifikus eseménypontok esetén nem célszerű a start/stop allokáció használata.

Ebben a feladatban az UIS és NIS tárolási stratégiáknál nem volt lényegi eltérés a futási eredményekben.

6.5. Esettanulmány III.

A harmadik esettanulmány alanya ([5] a másodikhoz nagyon hasonló, de multipurpose feladat, és a taszkok végrehajtási időben eltérnek.

Ebben a modellben a Sundaramoorthy2005 és a Mendez2003 modellek lettek összehasonlítva. A futási eredményeket az alábbi táblázatok tartalmazzák: a 6.8, 6.9, a 6.10, 6.11.

Értékelés. Érdekes módon mind UIS, mind NIS esetben a Sundaramoorthy2005 jobbnak bizonyult a precedencia-alapú Mendez2003-nál, még akkor sem, amikor az utóbbinál bekapcsoltuk az erősítő korlátozásokat. Az is megdöbbentő, hogy a Sundaramoorthy2005 az előző feladatban hamar elakadt, kis batchméretnél is, itt viszont kiváló eredményeket ért el, végigment a teszteseteken. A Mendez2003 ettől függetlenül továbbra is optimális makespan eredményeket adott, amelyeket a Sundaramoorthy2005 a kicsi intervallumszám miatt nem tudott elérni.

Mindezek arra engednek következtetni a II. esettanulmány eredményeivel együtt, hogy olyan tulajdonságok, mint a taszkok, berendezések, termékek száma, nem feltétlen jellemzik egy feladat bonyolultságát egy adott modell esetén. Ezeknél sokkal nagyobb hatása lehet a feladat struktúrájának (mul-

6.4. táblázat. Adatok a IV. esettanulmányhoz. Választási lehetőségek szintenként, berendezések és végrehajtási idők.

Termék	1. szint	2. szint	3. szint	4. szint	5. szint
A	(E1-6, 18.1)	(E7, 14), (E8-9, 5)	(E10-17, 12)	(E20-21, 9.5)	(E23-24, 24)
B	(E1-6, 23)	(E7, 14), (E8-9, 5)	(E10-19, 12)	(E22, 100)	(E25, 48)
C	(E1-6, 18.1)	(E7, 14), (E8-9, 5)	(E10-17, 24)	(E20-21, 9.3)	(E23-24, 24)
D	(E1-6, 20)	(E7, 11), (E8-9, 5)	(E10-17, 12)	(E20-21, 7.9)	(E23-24, 24)
E	(E1-6, 17)	(E7, 14), (E8-9, 5)	(E10-17, 12)	(E20-21, 12.5)	(E23-24, 24)

tipproduct és multipurpose), vagy a megadott paraméterek. A taszkok végrehajtási ideje sokkal változatosabb volt a III. esettanulmánynál.

6.6. Esettanulmány IV.

Egy nagyméretű festékipari feladatot vizsgáltunk [18] [16]. Az üzem ötféle terméket állít elő, egyenként öt szakaszban, amihez 25 berendezés áll rendelkezésre. A végrehajtandó taszkok rögzítettek, ám az egyes szinteken van választási lehetőség, hogy melyik legyen a végrehajtó berendezés. A 25 berendezés közül több egyforma - így összesen csak 9-féle van az üzemben. A végrehajtási idők nem tisztán egész értékűek, és széles skálán mozognak. A probléma adatait a 6.4 táblázat foglalja össze. Az eredeti feladathoz képest az egyszerűség kedvéért nem vettük figyelembe a berendezések beállításának idejét (setup time), valamint a tisztítási időket.

A feladatra az alábbi modelleket alkalmaztuk: Sundaramoorthy2005, Castro2004L, Castro2004K, S-gráf megoldó. A tárolási stratégia egyöntetűen NIS volt.

A feladat sajátosságai. Ez a multiproduct feladat több lényeges ponton is eltér az előző hárome settanulmánytól. A berendezések száma 25, de valójában csak 9 különböző van, ami jó alapot ad a lágy RTN (Castro2004L) és a kemény RTN (Castro2004K) reprezentáció összehasonlítására. Megjegyezzük, hogy az eddig megszokott feladatoknál a Castro2004L jobbnak bizonyult.

Más részről a feladatban van egy markáns szűk keresztmetszet, a második batch, aminek egy 100 óra hosszú taszkja van. Ez szemléletesen, globális időpontok esetén nagy számú átlapolódást követel meg, ami a start/stop modelleknek kedvezhet.

Futási eredmények és értékelés. A feladat futási eredményei a 6.12, a 6.13, a 6.14 táblázatokban.

Az S-gráf megoldó teljesítménye teljes kudarccal volt, nem sikerült tesztetést megoldania. Ez a szoftver hiányosságainak is betudható, mintsem az S-gráf módszertan korlátaira. Érdekes módon a start/stop allokációs Sundaramoorthy2005 is nagyon gyenge volt. A Castro2004 modell két implementációja volt a legsikeresebb, de közülük is a kemény RTN megvalósítás volt az egyértelműen erősebb.

Ez indokolja azt a hipotézisünket, hogy az irodalomban használatos, egyaránt RTN modellnek nevezett módszerek között is lehetnek különbségek, amelyek egy megfelelő feladat megoldásánál meg is nyilvánulnak.

6.5. táblázat. A kis feladat futási adatai

	Castro2004				Janak2004			
	cbc	eredmény	gurobi	eredmény	cbc	eredmény	gurobi	eredmény
2-TP4-D0	0.1	10	0	10	0.1	9	0	9
2-TP4-D1	0.1	10	0	10	0.1	9	0	9
2-TP4-D2	0.1	10	0	10	0.1	9	0	9
2-TP5-D0	0.1	10	0	10	0.8	8	0.1	8
2-TP5-D1	0.5	9	0	9	0.8	8	0.1	8
2-TP5-D2	0.5	8	0	8	0.8	8	0.1	8
2-TP6-D0	0.4	10	0	10	2.5	8	0.2	8
2-TP6-D1	0.9	9	0	9	2.5	8	0.2	8
2-TP6-D2	1.2	8	0	8	2.5	8	0.2	8

	Sundaramoorthy2005				Shaik2009			
	cbc	eredmény	gurobi	eredmény	cbc	eredmény	gurobi	eredmény
2-TP4	0	10	0	10	0	9	0	9
2-TP5	0.4	8	0	8	0.2	9	0	9
2-TP6	0.6	8	0	8	0.2	9	0	9

Mendez2003				
	cbc idő	eredmény	gurobi idő	eredmény
2	0	8	0	8

6.6. táblázat. A voudouris NIS feladat futási adatai

Voud-NIS	Castro2004			
	cbc idő	eredmény	gurobi idő	eredmény
2-2-2-2-TP10-D1	13.3	42	0.3	42
2-2-2-2-TP10-D2	21.1	42	0.5	42
2-2-2-2-TP11-D1	79	42	5.2	42
2-2-2-2-TP11-D2	171.2	41	8.5	41
3-2-2-2-TP11-D1	28.7	48	0.5	48
3-2-2-2-TP11-D2	42.6	48	3.5	48
3-2-2-2-TP12-D1	245	48	11.7	48
3-2-2-2-TP12-D2	511.9	47	18.8	47
3-3-2-2-TP12-D1	102.5	54	4	54
3-3-2-2-TP12-D2	169.7	53	5.8	53
3-3-2-2-TP13-D1	1000.4	54	70.4	53
3-3-2-2-TP13-D2	1000.6	53	150.9	53
3-3-3-2-TP13-D1	46.6	56	2.6	56
3-3-3-2-TP13-D2	193.2	56	11.9	56
3-3-3-2-TP14-D1	929.5	56	42.1	56
3-3-3-2-TP14-D2	1000.6	56	131	55
3-3-3-3-TP14-D1	109.2	58	3.7	58
3-3-3-3-TP14-D2	353.2	58	19.9	58
3-3-3-3-TP15-D1	1000.4	58	55.7	58
3-3-3-3-TP15-D2	1000.9	58	130.5	57
4-3-3-3-TP15-D1	511.6	64	10	64
4-3-3-3-TP15-D2	1000.6	64	43.8	64
4-3-3-3-TP16-D1	1000.6	64	423.3	64
4-3-3-3-TP16-D2	1000.7	63	580.9	63

Voud-NIS	Castro2004			
	cbc idő	eredmény	gurobi idő	eredmény
4-4-3-3-TP16-D1	1000.5	70	56	70
4-4-3-3-TP16-D2	1000.8	70	107	69
4-4-3-3-TP17-D1	1000.8	70	1000	70
4-4-3-3-TP17-D2	1000.9	69	1000	69
4-4-4-3-TP17-D1	1000.5	72	44.8	72
4-4-4-3-TP17-D2	1000.9	72	57.9	72
4-4-4-3-TP18-D1	1000.7	73	626.1	72
4-4-4-3-TP18-D2	1000.9	75	1000	71
4-4-4-4-TP18-D1	1000.5	74	26.3	74
4-4-4-4-TP18-D2	1001	77	114	74
4-4-4-4-TP19-D1	1000.6	75	411	74
4-4-4-4-TP19-D2	1001	74	1000	73
5-5-5-5-TP22-D1	1000.9	93	592.2	90
5-5-5-5-TP22-D2	1001.2	94	1000	90
5-5-5-5-TP23-D1	1000.8	93	1000	92
5-5-5-5-TP23-D2	1001.4	96	1000	91
6-6-6-6-TP26-D1	1001.1	111	1000	108
6-6-6-6-TP26-D2	1002	110	1000	107
6-6-6-6-TP27-D1	1001.6	109	1000	108
6-6-6-6-TP27-D2	1002	111	1000	107
7-7-7-7-TP30-D1	1001.5	128	1000	124
7-7-7-7-TP30-D2	1003.1	128	1000	123
7-7-7-7-TP31-D1	1002	131	1000	123
7-7-7-7-TP31-D2	1003	126	1000	124
8-8-8-8-TP34-D1	1004.2	145	1000	143
8-8-8-8-TP34-D2	1004.8	146	1000	138
8-8-8-8-TP35-D1	1001.8	145	1000	140
8-8-8-8-TP35-D2	1005.2	145	1000	140

	Sundaramoorthy2005				Shaik2010	
	cbc	eredmény	gurobi	eredmény	gurobi	eredmény
2-2-2-2-TP10	9.8	42	0.9	42	0.4	41
2-2-2-2-TP11	44.2	41	6.3	41	1.7	41
3-2-2-2-TP11	42.5	48	6.3	48	2.2	47
3-2-2-2-TP12	114.9	47	23.4	47	3.8	47
3-3-2-2-TP12	52	53	9.3	53	5.4	52
3-3-2-2-TP13	142.9	53	29.1	53	8.5	52
3-3-3-2-TP13	38.7	56	12.9	56	11.3	55
3-3-3-2-TP14	311.7	55	54.2	55	28.8	55
3-3-3-3-TP14	158.3	58	15.2	58	31.2	57
3-3-3-3-TP15	529.3	57	48.8	57	98.1	57
4-3-3-3-TP15	583.5	64	50.1	64	58.3	63
4-3-3-3-TP16	1000.5	63	97.4	63	183.3	63
4-4-3-3-TP16	1000.4	69	165.3	69	202.3	68
4-4-3-3-TP17	1000.5	69	447.3	68	372.8	68
4-4-4-3-TP17	959.5	72	51.1	72	340.4	71
4-4-4-3-TP18	1000.7	73	388.2	71	759	71
4-4-4-4-TP18	1000.7	74	123.1	74	691.7	73
4-4-4-4-TP19	1000.7	74	697.9	73	1000	73
5-5-5-5-TP22	1000.8	93	636.8	90		
5-5-5-5-TP23	1001.1	91	1000	89		
6-6-6-6-TP26	1000.9	114	1000	107		
6-6-6-6-TP27	1001.5	108	1000	106		
7-7-7-7-TP30	1001.2	126	1000	123		
7-7-7-7-TP31	1001.6	130	1000	124		
8-8-8-8-TP34	1001.4	143	1000	142		
8-8-8-8-TP35	1001.7	151	1000	139		

	Mendez2003				S-gráf	
	cbc idő	eredmény	gurobi idő	eredmény	futási idő	eredmény
2-2-2-2	0.5	41	0	41	0	41
3-2-2-2	0.7	46	0	46	0	46
3-3-2-2	1.5	52	0	52	0	52
3-3-3-2	3.5	55	0	55	0.1	55
3-3-3-3	6.6	57	0.1	56	0.2	57
4-3-3-3	7.3	62	0.1	62	0.2	62
4-4-3-3	12.5	67	0.1	67	0.1	67
4-4-4-3	13.7	71	0.2	71	1.7	71
4-4-4-4	27.9	73	0.8	73	2.9	73
5-5-5-5	1000.4	89	12.7	89	110.7	89
6-6-6-6			580.2	105	1000	105
7-7-7-7			1000	121	1000	121
8-8-8-8			1000	137	1000	167

6.7. táblázat. A voudoris UIS feladat futási adatai

	Castro2004			
	cbc idő	eredmény	gurobi idő	eredmény
2-2-2-2-TP10-D1	10.8	42	1.5	42
2-2-2-2-TP10-D2	23	42	1.6	42
2-2-2-2-TP11-D1	200.9	41	12.9	41
2-2-2-2-TP11-D2	195.8	40	11.8	40
3-2-2-2-TP11-D1	83.3	48	6.5	48
3-2-2-2-TP11-D2	112.5	48	7.4	48
3-2-2-2-TP12-D1	1000.4	47	82.7	47
3-2-2-2-TP12-D2	1000.6	47	86.4	47
3-3-2-2-TP12-D1	118.6	53	9.3	53
3-3-2-2-TP12-D2	285.8	53	15.5	53
3-3-2-2-TP13-D1	1000.5	53	158.5	53
3-3-2-2-TP13-D2	1000.8	52	306.8	52
3-3-3-2-TP13-D1	755.1	56	12.2	56
3-3-3-2-TP13-D2	614.2	56	25.3	56
3-3-3-2-TP14-D1	1000.6	56	219.5	56
3-3-3-2-TP14-D2	1000.7	55	256.1	54

	Castro2004			
	cbc idő	eredmény	gurobi idő	eredmény
3-3-3-3-TP14-D1	249	57	15.4	57
3-3-3-3-TP14-D2	330.4	57	12.5	57
3-3-3-3-TP15-D1	1000.5	57	112.9	57
3-3-3-3-TP15-D2	1000.8	57	303.4	55
4-3-3-3-TP15-D1	1000.4	64	56.4	63
4-3-3-3-TP15-D2	1000.5	63	77	63
4-3-3-3-TP16-D1	1000.5	64	1000	63
4-3-3-3-TP16-D2	1000.7	62	1000	62
4-4-3-3-TP16-D1	1000.7	69	121.9	68
4-4-3-3-TP16-D2	1000.6	68	118.4	68
4-4-3-3-TP17-D1	1000.6	70	1000	69
4-4-3-3-TP17-D2	1000.8	69	1000	68
4-4-4-3-TP17-D1	1000.7	72	170.4	71
4-4-4-3-TP17-D2	1000.9	72	159.5	71
4-4-4-3-TP18-D1	1000.7	75	1000	72
4-4-4-3-TP18-D2	1000.8	73	1000	71
4-4-4-4-TP18-D1	1000.6	73	132.8	73
4-4-4-4-TP18-D2	1000.6	75	115.1	72
4-4-4-4-TP19-D1	1000.8	74	1000	73
4-4-4-4-TP19-D2	1000.9	76	1000	71
5-5-5-5-TP22-D1	1000.9	91	328.6	88
5-5-5-5-TP22-D2	1001.2	91	348.7	87
5-5-5-5-TP23-D1	1001.3	95	1000	90
5-5-5-5-TP23-D2	61.9	112	1000	87

	Sundaramoorthy2005			
	cbc idő	eredmény	gurobi idő	eredmény
2-2-2-2-TP10	59.6	42	3	42
2-2-2-2-TP11	282.5	40	5.3	40
3-2-2-2-TP11	245.3	48	11.3	48
3-2-2-2-TP12	719.3	47	30.9	47
3-3-2-2-TP12	130.6	53	49.8	53
3-3-2-2-TP13	1000.6	53	16.9	52
3-3-3-2-TP13	672.5	56	29.6	56
3-3-3-2-TP14	1000.4	54	64.2	54
3-3-3-3-TP14	1000.4	57	44.4	57
3-3-3-3-TP15	1000.6	56	86	55
4-3-3-3-TP15	1000.6	65	134.4	63
4-3-3-3-TP16	1000.7	65	430.9	62
4-4-3-3-TP16	1000.6	69	345.7	68
4-4-3-3-TP17	1000.7	68	429	67
4-4-4-3-TP17	1000.8	73	355.2	71
4-4-4-3-TP18	1000.8	71	677.8	69
4-4-4-4-TP18	1000.9	75	375.2	72
4-4-4-4-TP19	1000.9	80	740.7	70
5-5-5-5-TP22	1001	94	1000	88
5-5-5-5-TP23	1001.2	97	1000	88
6-6-6-6-TP26	1001.4	113	1000	103
6-6-6-6-TP27	1001.2	111	1000	107
7-7-7-7-TP30	1001.9	125	1000	123
7-7-7-7-TP31	1002.1	131	1000	122
8-8-8-8-TP34	1001.6	147	1000	140
8-8-8-8-TP35	1002.3	144	1000	142

	Mendez2003	
	futási idő	eredmény
2-2-2-2	0	39
3-2-2-2	0	46
3-3-2-2	0	52
3-3-3-2	0.1	52
3-3-3-3	0.1	53
4-3-3-3	0.1	60
4-4-3-3	0.1	67
4-4-4-3	0.2	67
4-4-4-4	8.2	68
5-5-5-5	30.6	83
6-6-6-6	51.9	97
7-7-7-7	84.5	112
8-8-8-8	517.7	127

6.8. táblázat. A ferrer NIS feladat futási adatai

	Sundaramoorthy2005			
	cbc idő	eredmény	gurobi idő	eredmény
2-2-2-2-TP8	2.7	88	0.1	88
2-2-2-2-TP9	8.7	87	0.4	87
3-2-2-2-TP10	12.7	96	0.7	96
3-2-2-2-TP9	3.7	97	0.2	97
3-3-2-2-TP10	4.9	114	0.2	114
3-3-2-2-TP11	21	113	1.1	113
3-3-3-2-TP10	0.8	129	0.2	129
3-3-3-2-TP11	15	119	0.7	119
3-3-3-3-TP11	5.6	129	0.3	129
3-3-3-3-TP12	21.7	128	1.5	128
4-3-3-3-TP12	4.7	138	0.4	138
4-3-3-3-TP13	28	137	7.2	137
4-4-3-3-TP13	8.5	155	0.6	155
4-4-3-3-TP14	117.4	154	15.1	154
4-4-4-3-TP13	1	170	0.3	170
4-4-4-3-TP14	31.4	160	2.6	160
4-4-4-4-TP14	6.1	170	0.4	170
4-4-4-4-TP15	71.3	169	11	169
5-5-5-5-TP17	10	211	0.5	211
5-5-5-5-TP18	211.9	210	29	210
6-6-6-6-TP20	12.7	252	1	252
6-6-6-6-TP21	432	251	55.7	251
7-7-7-7-TP23	7.2	293	1.1	293
7-7-7-7-TP24	622.4	292	86.9	292
8-8-8-8-TP26	17.4	334	1.7	334
8-8-8-8-TP27	1001.9	333	104.5	333

6.9. táblázat. A ferrer feladat futási eredményei

	Mendez2003				Mendez2003 – multibatch			
	cbc	eredmény	gurobi	eredmény	cbc	eredmény	gurobi	eredmény
2-2-2-2	5	87	0.1	87	0.6	87	0	87
3-2-2-2	16.6	96	0.8	96	1.3	96	0	96
3-3-2-2	78.9	109	2.7	109	1.9	109	0.1	109
3-3-3-2	657.5	115	6.4	115	3	115	0.2	115
3-3-3-3	1000	125	87.8	125	10.7	125	0.4	125
4-3-3-3	1000	134	1000	134	30.7	134	0.8	134
4-4-3-3	1000	149	1000	149	78.8	149	2.2	149
4-4-4-3	1000	155	1000	155	214.8	155	4.5	155
4-4-4-4	1000	165	1000	165	685.5	165	12.9	165
5-5-5-5	1000	218	1000	204	1001.1	204	490	204
6-6-6-6	1000	277	1000	246	1001.3	257	1000	243
7-7-7-7	1000	356	1000	291	1001	306	1000	290
8-8-8-8	1000	404	1000	334	1001	363	1000	324

6.10. táblázat. A ferrer UIS feladat futási eredményei

Sundaramoorth2005				
	cbc	eredmény	gurobi	eredmény
2-2-2-2-TP8	7.1	84	0.3	84
2-2-2-2-TP9	20.8	82	1.9	82
3-2-2-2-TP10	49.1	91	2.6	91
3-2-2-2-TP9	16.0	93	0.7	93
3-3-2-2-TP10	25.2	110	1.5	110
3-3-2-2-TP11	110.0	108	3.4	108
3-3-3-2-TP10	18.0	112	1.0	112
3-3-3-2-TP11	123.3	110	5.0	110
3-3-3-3-TP11	157.5	121	6.6	121
3-3-3-3-TP12	495.1	119	20.2	119
4-3-3-3-TP12	868.9	130	7.1	130
4-3-3-3-TP13	1000.4	128	19.3	128
4-4-3-3-TP13	504.9	147	11.5	147
4-4-3-3-TP14	1000.6	145	45.3	145
4-4-4-3-TP13	276.7	149	6.4	149
4-4-4-3-TP14	1000.5	147	46.2	147
4-4-4-4-TP14	1000.4	158	61.1	158
4-4-4-4-TP15	1000.6	156	271.2	156
5-5-5-5-TP17	1000.7	195	278.1	195
5-5-5-5-TP18	1001.5	193	1000.0	193
6-6-6-6-TP20	1001.2	232	1000.0	232
6-6-6-6-TP21	1001.0	232	1000.0	230
7-7-7-7-TP23	1001.2	269	1000.0	269
7-7-7-7-TP24	1002.6	270	1000.0	267
8-8-8-8-TP26	1001.6	306	1000.0	306
8-8-8-8-TP27	1002.2	307	1000.0	304

6.11. táblázat. A ferrer UIS feladat futási eredményei

	Mendez2003			
	cbc	eredmény	gurobi	eredmény
2-2-2-2	0.5	80	0	80
3-2-2-2	1.8	89	0	89
3-3-2-2	2.6	104	0.1	104
3-3-3-2	5.2	104	0.4	104
3-3-3-3	16.4	115	0.3	115
4-3-3-3	96.7	124	0.5	124
4-4-3-3	775.3	139	0.7	139
4-4-4-3	1000.1	139	6.2	139
4-4-4-4	1001.9	150	3.4	150
5-5-5-5	1001.9	185	415.8	185
6-6-6-6	198.6	220	1000	220
7-7-7-7	1000.2	143	1000.1	143

6.12. táblázat. A festékipari feladat futási eredményei

	Castro2004	
	gurobi idő	megoldás
2-1-2-2-2-TP8-D2	1.2	194
2-1-2-2-2-TP9-D2	28.4	188
3-1-3-2-2-TP10-D2	43.1	192.6
3-1-3-2-2-TP9-D2	9.1	195.1
4-1-4-3-3-TP11-D2	37.1	242.5
4-1-4-3-3-TP12-D2	418.8	240.6
4-1-4-4-4-TP12-D2	32.3	266.5
4-1-4-4-4-TP13-D2	1000	264.6

6.13. táblázat. A festékipari feladat futási eredményei

	Castro2004 heurisztikák nélkül	
batchnum	gurobi idő	
2-1-2-2-2-TP8-D2	3.1	194
2-1-2-2-2-TP9-D2	5.6	188
3-1-3-2-2-TP10-D2	18.2	194.5
3-1-3-2-2-TP9-D2	7	195.6
4-1-4-3-3-TP11-D2	22.8	243.6
4-1-4-3-3-TP12-D2	179.7	242.5
4-1-4-4-4-TP12-D2	30.2	267.6
4-1-4-4-4-TP13-D2	658.4	266.5
4-2-4-4-4-TP12-D2	35.8	295.6
4-2-4-4-4-TP13-D2	160.2	294.5
5-2-5-5-5-TP14-D2	87.7	343.6
5-2-5-5-5-TP15-D2	822.1	342.5
5-3-5-5-5-TP14-D2	67.1	394
5-3-5-5-5-TP15-D2	176.7	388
6-3-6-6-6-TP16-D2	150.8	419.6
6-3-6-6-6-TP17-D2	741	418.5

6.14. táblázat. A festékipari feladat futási eredményei

	Sundaramoorthy2005	
	gurobi idő	megoldás
2-1-2-2-2-TP8	40.9	194
2-1-2-2-2-TP9	490.7	188
3-1-3-2-2-TP9	78.6	194
4-1-4-3-3-TP11	1000	215.1

7. fejezet

Összefoglalás

Dolgozatunkban szakaszos üzemű vegyipari rendszerek üzemeltetése során fellépő ütemezési problémák megoldására kidolgozott módszereket vizsgáltunk. Bemutattuk az előforduló feladatok több szempont alapján történő osztályozását, majd ismertettük az irodalomban ezen feladatosztályok megoldására kidolgozott módszereket azok előnyeit, hátrányait hangsúlyozva.

Munkánk fő célja ezen módszerek széleskörű tesztelése volt, melynek első lépéseként az irodalomban publikált feladatok és modellek gyűjtésére és kategorizálására alkalmas online adatbázist készítettünk. Az utóbbi 10 évben megjelent irodalmi példák és esettanulmányok összegyűjtését követően kiválasztottunk néhány feladatot, melyeken a state-of-the-art megoldó módszerek viselkedését elemeztük. A különböző matematikai modelleken alapuló megközelítések leírására egy egységes formát alkottunk meg, melyben ezeket implementáltuk. Empirikus vizsgálatok során ezeket összehasonlítottuk.

A több mint háromezer futási eredményből hasznos következtetéseket vontunk le arra vonatkozólag, hogy adott feladattípus esetén mely módszerek milyen megoldó algoritmusának alkalmazása előnyös.

A munka folytatásaként szeretnénk bővíteni a feladat és modelladatbázist. Célunk továbbá a modellezési aspektusok széleskörű összehasonlítása. Távlati célunk egy intelligens ütemező keretrendszer fejlesztése, mely a feladat tulajdonságai alapján képes javaslatot tenni a legmegfelelőbb megoldó módszerre.

A. Függelék

Futási eredmények

A.1. táblázat. Mendez2003 modell eredmények

név	GLPK alapbeállítás	GLPK cover	GLPK mir	GLPK gomory	GLPK clique	GLPK cuts	CBC	CPLEX	Gurobi
0-10	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.1 131	100.0 131	100.0 131
0-2	0.0 27	0.0 27	0.0 27	0.0 27	0.0 27	0.0 27	0.0 27	0.0 27	0.0 27
0-3	0.0 40	0.0 40	0.0 40	0.0 40	0.0 40	0.0 40	0.2 40	0.0 40	0.0 40
0-4	0.1 53	0.1 53	0.3 53	0.1 53	0.1 53	0.3 53	0.8 53	0.0 53	0.0 53
0-5	1.3 66	1.4 66	5.5 66	1.3 66	1.3 66	5.6 66	3.6 66	0.4 66	0.1 66
10-10	100.1 –	100.1 –	100.1 –	100.1 –	100.1 –	100.1 –	100.1 170	100.0 157	100.0 157
10-2	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.0 59	100.0 55	100.0 55
10-5	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.1 101	100.0 93	100.0 93
10-7	100.0 –	100.0 –	100.0 –	100.0 –	100.1 –	100.0 –	100.1 126	100.0 118	100.0 118
12-12	100.2 –	100.2 –	100.3 –	100.2 –	100.2 –	100.2 –	100.2 196	100.0 192	100.0 189
1-0	0.0 5	0.0 5	0.0 5	0.0 5	0.0 5	0.0 5	0.0 5	0.0 5	0.0 5
1-1	0.0 19	0.0 19	0.0 19	0.0 19	0.0 19	0.0 19	0.0 19	0.0 19	0.0 19
2-0	0.0 8	0.0 8	0.0 8	0.0 8	0.0 8	0.0 8	0.0 8	0.0 8	0.0 8
2-1	0.0 21	0.0 21	0.0 21	0.0 21	0.0 21	0.0 21	0.0 21	0.0 21	0.0 21
2-2	0.0 31	0.0 31	0.1 31	0.0 31	0.0 31	0.1 31	0.2 31	0.0 31	0.0 31
2-3	0.2 44	0.2 44	0.6 44	0.2 44	0.2 44	0.6 44	1.0 44	0.1 44	0.0 44
30-0	100.3 –	100.3 –	100.3 –	100.3 –	100.3 –	100.3 –	100.3 92	100.1 92	100.0 92
3-0	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11	0.1 11	0.0 11	0.0 11
3-1	0.0 24	0.0 24	0.0 24	0.0 24	0.0 24	0.0 24	0.1 24	0.0 24	0.0 24
3-2	0.1 34	0.1 34	0.4 34	0.1 34	0.1 34	0.4 34	0.7 34	0.1 34	0.0 34
3-3	1.5 47	1.5 47	6.4 47	1.5 47	1.5 47	6.5 47	3.5 47	0.5 47	0.1 47
3-6	100.0 98	100.0 98	100.0 98	100.0 98	100.0 98	100.0 98	101.2 86	47.1 86	30.0 86
4-0	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14	0.2 14	0.0 14	0.0 14
4-1	0.1 27	0.1 27	0.2 27	0.1 27	0.1 27	0.2 27	0.5 27	0.1 27	0.0 27
4-4	100.0 62	100.0 62	100.0 63	100.0 62	100.0 62	100.0 63	87.7 62	2.5 62	1.4 62
5-0	0.1 17	0.1 17	0.2 17	0.1 17	0.1 17	0.2 17	0.6 17	0.0 17	0.0 17
5-1	0.6 30	0.6 30	2.4 30	0.6 30	0.6 30	2.4 30	1.7 30	0.3 30	0.1 30
5-3	100.0 53	100.0 53	100.0 53	100.0 53	100.0 53	100.0 53	45.4 53	2.0 53	1.1 53
5-5	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.0 –	100.1 78	100.0 78	100.0 78
6-3	100.0 61	100.0 61	100.0 61	100.0 61	100.0 61	100.0 61	100.1 56	9.5 56	10.7 56
7-10	100.1 –	100.1 –	100.1 –	100.1 –	100.1 –	100.1 –	100.1 150	100.0 148	100.0 148

A.2. táblázat. Castro2004 modell eredmények 2.

név	GLPK alapbeállítás	GLPK cover	GLPK mir	GLPK gomory	GLPK clique	GLPK cuts	CBC	CPLEX	Gurobi
3-0-p4	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
3-0-p5	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11	0.0 11
3-1-p5	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
3-1-p6	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
3-1-p7	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
3-2-p8	0.1 25	0.1 25	0.2 25	0.2 25	0.1 25	0.4 25	1.2 25	0.1 25	0.2 25
3-3-p10	2.0 32	2.1 32	4.4 32	2.0 32	2.1 32	4.1 32	2.8 32	0.3 32	0.5 32
3-3-p16	100.4 31	100.4 31	100.4 31	100.4 31	100.4 31	100.4 31	100.2 32	100.0 30	100.0 30
3-3-p8	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
3-3-p9	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
3-6-p12	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.0 –
3-6-p14	0.2 –	0.2 –	0.2 –	0.2 –	0.2 –	0.2 –	0.1 –	0.1 –	0.1 –
3-6-p15	0.3 –	0.3 –	0.3 –	0.3 –	0.3 –	0.3 –	0.1 –	0.2 –	0.1 –
3-6-p16	100.4 55	100.4 55	100.4 55	100.4 55	100.4 55	100.4 55	3.8 55	1.6 55	61.0 55
3-6-p25	102.8 56	102.6 56	102.7 56	102.7 55	102.7 56	102.6 55	100.8 56	100.2 54	100.0 54
4-0-p6	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14	0.0 14
4-0-p9	0.6 14	0.6 14	1.4 14	1.9 14	0.6 14	4.6 14	2.4 14	0.1 14	0.2 14
4-1-p12	100.1 20	100.1 20	100.1 20	100.1 20	100.1 20	100.1 20	93.6 20	7.5 20	8.2 20
4-1-p6	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
4-1-p7	0.0 22	0.0 22	0.0 22	0.0 22	0.0 22	0.0 22	0.4 22	0.1 22	0.0 22
4-4-p10	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –	0.0 –
4-4-p12	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.0 –	0.1 –	0.1 –
4-4-p13	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.1 –	0.0 –
4-4-p21	101.0 41	101.0 41	101.0 41	101.1 42	101.0 41	101.1 42	100.5 42	100.0 40	100.0 41

A.3. táblázat. Castro2004 modell eredmények 3.

név	GLPK alapbeállítás		GLPK cover		GLPK mir		GLPK gomory		GLPK clique		GLPK cuts		CBC		CPLEX		Gurobi	
5-0-p11	8.9	17	9.1	17	18.6	17	44.9	17	9.2	17	67.4	17	–	17	1.2	17	1.1	17
5-0-p6	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–
5-0-p7	0.0	17	0.0	17	0.0	17	0.0	17	0.0	17	0.0	17	0.0	17	0.0	17	0.0	17
5-1-p14	100.2	23	100.2	23	100.2	23	100.2	23	100.2	23	100.2	24	100.2	23	52.8	23	99.4	23
5-1-p7	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–
5-1-p8	0.0	25	0.0	25	0.0	25	0.0	25	0.0	25	0.1	25	0.5	25	0.1	25	0.1	25
5-1-p9	0.3	24	0.3	24	0.4	24	0.8	24	0.3	24	1.1	24	2.0	24	0.2	24	0.1	24
5-3-p10	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–	0.0	–
5-3-p11	0.1	–	0.1	–	0.1	–	0.1	–	0.1	–	0.1	–	0.0	–	0.1	–	0.0	–
5-3-p12	26.1	37	26.4	37	35.6	37	16.9	37	26.8	37	46.3	37	13.4	37	1.7	37	3.1	37
5-3-p15	0.3	–	0.3	–	0.3	–	0.3	–	0.3	–	0.3	–	0.1	–	0.2	–	0.1	–
5-3-p20	100.8	36	100.8	36	100.8	37	100.8	37	100.8	36	100.8	38	100.3	37	100.0	35	100.0	36
5-5-p16	100.4	52	100.4	52	100.4	52	100.4	52	100.4	52	100.4	52	52.5	52	24.7	52	36.0	52
5-5-p26	102.7	52	102.7	52	102.6	52	102.7	54	102.7	52	103.2	54	101.1	52	100.0	50	100.0	51
6-3-p12	0.1	–	0.1	–	0.1	–	0.1	–	0.1	–	0.1	–	0.0	–	0.1	–	0.1	–
6-3-p13	92.0	40	93.2	40	100.2	41	68.3	40	94.3	40	84.0	40	23.2	40	2.8	40	5.3	40
6-3-p22	101.3	40	101.3	40	101.2	40	101.3	40	101.3	40	101.2	40	100.4	41	100.0	38	100.0	39
7-10-p27	3.1	–	3.1	–	3.1	–	3.1	–	3.2	–	3.1	–	1.0	–	1.3	–	0.6	–
7-10-p28	103.9	96	103.8	96	103.9	97	103.9	96	103.9	96	104.0	96	59.5	95	12.8	95	100.0	95
7-10-p43	65.4	–	65.1	–	65.8	–	131.8	–	64.9	–	131.5	–	105.6	114	100.0	98	100.1	95

Irodalomjegyzék

- [1] C. A. Méndez, J. Cerda, I. E. Grossmann, I. Harjunkoski, and M. Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers & Chemical Engineering*, 30(6-7):913–946, May 2006.
- [2] E. Sanmarti, T. Holczinger, L. Puigjaner, and F. Friedler. Combinatorial framework for effective scheduling of multipurpose batch plants. *AIChE Journal*, 48(11):2557–2570, 2002.
- [3] Tibor Holczinger. *Method for scheduling non-intermediate storage batch process systems*. PhD thesis, Doctoral School of Information Science and Technology - University of Veszprem, 2004.
- [4] E. Kondili, C. C. Pantelides, and R. W. H. Sargent. A general algorithm for short-term scheduling of batch operations—i. milp formulation. *Computers & Chemical Engineering*, 17(2):211–227, February 1993.
- [5] S. Ferrer-Nadal, E. Capón-García, C. A. Méndez, and L. Puigjaner. Material transfer operations in batch scheduling. a critical modeling issue. *Industrial & Engineering Chemistry Research*, 47:7721–7732, 2008.
- [6] M. Hegyhati, T. Majozi, T. Holczinger, and F. Friedler. Practical infeasibility of cross-transfer in batch plants with complex recipes: S-graph vs milp methods. *Chemical Engineering Science*, 64(3):605–610, February 2009.
- [7] C. C. Pantelides. Unified frameworks for optimal process planning and scheduling. In Rip pin D. W. T., Hale J. C., and Davis J., editors, *Proceedings of the second international conference on foundations of computer-aided process operations*, pages 253–274, 1993.
- [8] C. T. Maravelias and I. E. Grossmann. A new continuous-time state task network formulation for short term scheduling of multipurpose batch plants. *Computer Aided Chemical Engineering*, 14:215–220, 2003.

- [9] P. M. Castro, A. P. Barbosa-Póvoa, H. A. Matos, and Q. A. Novais. Simple continuous-time formulation for short-term scheduling of batch and continuous processes. *Industrial & Engineering Chemistry Research*, 43(1):105–118, 2004.
- [10] Arul Sundaramoorthy and I.A. Karimi. A simpler better slot-based continuous-time formulation for short-term scheduling in multipurpose batch plants. *Chemical Engineering Science*, 60(10):2679 – 2702, 2005.
- [11] M. G. Ierapetritou and C. A. Floudas. Short term scheduling: New mathematical models vs algorithmic improvements. *Computers & Chemical Engineering*, 22:S419–S426, 1998.
- [12] Stacy L. Janak, Xiaoxia Lin, and Christodoulos A. Floudas. Enhanced continuous-time unit-specific event-based formulation for short-term scheduling of multipurpose batch processes: Resource constraints and mixed storage policies. *Industrial & Engineering Chemistry Research*, 44(2):426–426, 2005.
- [13] M. A. Shaik, C. A. Floudas, J. Kallrath, and H. J. Pitz. Production scheduling of a large-scale industrial continuous plant: Short-term and medium-term scheduling. *Computers & Chemical Engineering*, 33(3):670–686, March 2009.
- [14] J. Cerda, G. P. Henning, and I. E. Grossmann. A mixed-integer linear programming model for short-term scheduling of single-stage multiproduct batch plants with parallel lines. *Industrial & Engineering Chemistry Research*, 36(5):1695–1707, 1997.
- [15] C. A. Méndez, G. P. Henning, and J. Cerda. Optimal scheduling of batch plants satisfying multiple product orders with different due-dates. *Computers & Chemical Engineering*, 24(9-10):2223–2245, October 2000.
- [16] C. A. Méndez, G. P. Henning, and J. Cerda. An milp continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. *Computers & Chemical Engineering*, 25(4-6):701–711, May 2001.
- [17] C. A. Méndez and J. Cerda. An milp continuous-time framework for short-term scheduling of multipurpose batch processes under different operation strategies. *Optimization and Engineering*, 4:7–22, 2003.
- [18] J. M. Pinto and I. E. Grossmann. A continuous time mixed integer linear programming model for short term scheduling of multistage batch

- plants. *Industrial & Engineering Chemistry Research*, 34(9):3037–3051, 1995.
- [19] S. B. Kim, H. K. Lee, I. B. Lee, E. S. Lee, and B. Lee. Scheduling of non-sequential multipurpose batch processes under finite intermediate storage policy. *Computers & Chemical Engineering*, 24(2-7):1603–1610, July 2000.
 - [20] T. Majozzi and X. X. Zhu. A novel continuous-time milp formulation for multipurpose batch plants. 1. short-term scheduling. *Industrial & Engineering Chemistry Research*, 40(25):5935–5949, 2001.
 - [21] E. Sanmarti, F. Friedler, and L. Puigjaner. Combinatorial technique for short term scheduling of multipurpose batch plants based on schedule-graph representation. *Computer Aided Chemical Engineering*, 22:S847–850, 1998.
 - [22] T. Holczinger, J. Romero, L. Puigjaner, and F. Friedler. Scheduling of multipurpose batch processes with multiple batches of the products. *Hungarian Journal of Industrial Chemistry*, 30:305–312, 2002.
 - [23] M. Ghaeli, P. A. Bahri, P. Lee, and T. Gu. Petri-net based formulation and algorithm for short-term scheduling of batch plants. *Computers & Chemical Engineering*, 29(2):249–259, January 2005.
 - [24] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, April 1994.
 - [25] S. Panek, S. Engell, S. Subbiah, and O. Stursberg. Scheduling of multi-product batch plants based upon timed automata models. *Computers & Chemical Engineering*, 32(1-2):275–291, January 2008.
 - [26] J. Smidla and I. Heckl. S-graph based parallel algorithm to the scheduling of multipurpose batch plants. *Chemical Engineering Transactions*, 21:937–942, 2010.
 - [27] Tóth Benjámín Éles András, Kovács Balázs. Szakaszos üzemu termelo rendszerek ütemezése: feladatosztályok és megoldó módszerek. 2011.
 - [28] V. T. Voudouris and I. E. Grossmann. Milp model for scheduling and design of a special class of multipurpose batch plants. *Computers & Chemical Engineering*, 20(11):1335–1360, 1996.